# Open Call-by-Value

Beniamino Accattoli[1] and Giulio Guerrieri[2]

[1] INRIA, UMR 7161, LIX, École Polytechnique, beniamino.accattoli@inria.fr,
[2] Aix Marseille Université, I2M UMR 7373, giulio.guerrieri@univ-amu.fr

**Abstract.** The elegant theory of the call-by-value lambda-calculus relies on weak evaluation and closed terms, that are natural hypotheses in the study of programming languages. To model proof assistants, however, strong evaluation and open terms are required, and it is well known that the operational semantics of call-by-value becomes problematic in this case. Here we study the intermediate setting—that we call Open Call-by-Value—of weak evaluation with open terms, on top of which Grégoire and Leroy designed the abstract machine of Coq. Various calculi for Open Call-by-Value already exist, each one with its pros and cons. This paper presents a detailed comparative study of the operational semantics of four of them, coming from different areas such as the study of abstract machines, denotational semantics, linear logic proof nets, and sequent calculus. We show that these calculi are all equivalent from a termination point of view, justifying the slogan Open Call-by-Value.

## 1 Introduction

Plotkin's call-by-value $\lambda$-calculus [27] is at the heart of programming languages such as Ocaml and proof assistants such as Coq. In the study of programming languages, call-by-value (CBV) evaluation is usually *weak*, *i.e.* it does not reduce under abstractions, and terms are assumed to be *closed*. These constraints give rise to a beautiful theory—let us call it *Closed CBV*—having the following *harmony property*, that relates rewriting and normal forms:

*Closed normal forms are values* (and values are normal forms)

where *values* are variables and abstractions. Harmony expresses a form of internal completeness with respect to unconstrained $\beta$-reduction: the restriction to CBV $\beta$-reduction (referred to as *$\beta_v$-reduction*, according to which a $\beta$-redex can be fired only when the argument is a value) has an impact on the order in which redexes are evaluated, but evaluation never gets stuck, as every $\beta$-redex will eventually become a $\beta_v$-redex and be fired, unless evaluation diverges.

It often happens, however, that one needs to go beyond the perfect setting of Closed CBV, by considering *Strong CBV*, where reduction under abstractions is allowed and terms may be open, or the intermediate setting of *Open CBV*, where evaluation is weak but terms are not necessarily closed. The need arises, most notably, when trying to describe the implementation model of Coq [13], but also from other motivations, as denotational semantics [26,29,3,8], monad and CPS translations and the associated equational theories [22,30,31,12,17], bisimulations [19], partial evaluation [18], linear logic proof nets [2], or cost models [1].

*Naïve Open CBV.* In call-by-name (CBN) turning to open terms or strong evaluation is harmless because CBN does not impose any special form to the arguments of $\beta$-redexes. On the contrary, turning to Open or Strong CBV is delicate. If one simply considers Plotkin's weak $\beta_v$-reduction on open terms—let us call it *Naïve Open CBV*—then harmony does no longer hold, as there are open $\beta$-normal forms that are not values, *e.g.* $xx$, $x(\lambda y.y)$, $x(yz)$ or $xyz$. As a consequence, there are *stuck $\beta$-redexes* such as $(\lambda y.t)(xx)$, *i.e.* $\beta$-redexes that will never be fired because their argument is normal, but it is not a value, nor will it ever become one. Such stuck $\beta$-redexes are a disease typical of (Naïve) Open CBV, but they spread to Strong CBV as well (also in the closed case), because evaluating under abstraction forces to deal with locally open terms: *e.g.* the variable $x$ is locally open with respect to $(\lambda y.t)(xx)$ in $s = \lambda x.((\lambda y.t)(xx))$.

The real issue with stuck $\beta$-redexes is that they prevent the creation of other redexes, and provide *premature* $\beta_v$-normal forms. The issue is serious, as it can affect termination, and thus impact on notions of observational equivalence. Let $\delta \coloneqq \lambda x.(xx)$. The problem is exemplified by the terms $t$ and $u$ in Eq. (1) below.

$$t \coloneqq ((\lambda y.\delta)(zz))\delta \qquad\qquad u \coloneqq \delta((\lambda y.\delta)(zz)) \tag{1}$$

In Naïve Open CBV, $t$ and $u$ are premature $\beta_v$-normal forms because they both have a stuck $\beta$-redex forbidding evaluation to keep going, while one would expect them to behave like the divergent term $\Omega \coloneqq \delta\delta$ (see [26,29,3,2,8,15] and pp. 7-12).

*Open CBV.* In his seminal work, Plotkin already pointed out an asymmetry between CBN and CBV: his CPS translation is sound and complete for CBN, but only sound for CBV. This fact led to a number of studies about monad, CPS, and logical translations [22,30,31,21,12,17] that introduced many proposals of improved calculi for CBV. Starting with the seminal work of Paolini and Ronchi Della Rocca [26,24,29], the dissonance between open terms and CBV has been repeatedly pointed out and studied *per se* via various calculi [13,3,2,8,15,14,1]. A further point of view on CBV comes from the computational interpretation of sequent calculus due to Curien and Herbelin [9]. An important point is that the focus of most of these works is on Strong CBV.

These solutions inevitably extend $\beta_v$-reduction with some other rule(s) or constructor (as `let`-expressions) to deal with stuck $\beta$-redexes, or even go as far as changing the applicative structure of $\lambda$-terms, as in the sequent calculus approach. They arise from different perspectives and each one has its pros and cons. By design, these calculi (when looked at in the context of Open CBV) are never observationally equivalent to Naïve Open CBV, as they all manage to (re)move stuck $\beta$-redexes and may diverge when Naïve Open CBV is instead stuck. Each one of these calculi, however, has its own notion of evaluation and normal form, and their mutual relationships are not evident.

The aim of this paper is to draw the attention of the community on Open CBV. We believe that it is somewhat deceiving that the mainstream operational theory of CBV, however elegant, has to rely on closed terms, because it restricts the modularity of the framework, and raises the suspicion that the true essence of CBV has yet to be found. There is a real gap, indeed, between Closed and

Strong CBV, as Strong CBV cannot be seen as an iteration of Closed CBV under abstractions because such an iteration has to deal with open terms. To improve the implementation of Coq [13], Grégoire and Leroy see Strong CBV as the iteration of the intermediate case of Open CBV, but they do not explore its theory. Here we exalt their point of view, providing a thorough operational study of Open CBV. We insist on Open CBV rather than Strong CBV because:

1. Stuck $\beta$-redexes and premature $\beta_v$-normal forms are already in Open CBV;
2. Open CBV has a simpler rewriting theory than Strong CBV;
3. Our previous studies of Strong CBV in [3] and [8] naturally organized themselves as properties of Open CBV that were lifted to Strong CBV by a simple iteration under abstractions.

Our contributions are along two axes:

1. *Termination Equivalence of the Proposals*: we show that the proposed generalizations of Naïve Open CBV are equivalent, in the sense that they have exactly the same sets of normalizing and diverging $\lambda$-terms. Therefore, *there is just one notion of Open CBV*, independently of its specific syntactic incarnation.
2. *Quantitative Analyses and Cost Models*: the termination results are complemented with quantitative analyses establishing precise relationships between the number of steps needed to evaluate a given term in the various calculi. In particular, we relate the cost models of the various proposals.

*Equivalence of the Proposals.* We focus on four proposals for Open CBV, as other solutions, *e.g.* Moggi's [22] or Herbelin and Zimmerman's [17], are already known to be equivalent to these ones (see the end of Sect. 2):

1. *The Fireball Calculus* $\lambda_{\mathtt{fire}}$, that extends values to *fireballs* by adding so-called *inert terms* in order to restore harmony—it was introduced without a name by Paolini and Ronchi Della Rocca [26,29], then rediscovered independently first by Leroy and Grégoire [13] to improve the implementation of Coq, and then by Accattoli and Sacerdoti Coen [1] to study cost models;
2. *The Value Substitution Calculus* $\lambda_{\mathtt{vsub}}$, coming from the linear logic interpretation of CBV and using explicit substitutions and contextual rewriting rules to circumvent stuck $\beta$-redexes—it was introduced by Accattoli and Paolini [3] and it is a graph-free presentation of proof nets for the CBV $\lambda$-calculus [2];
3. *The Shuffling Calculus* $\lambda_{\mathtt{sh}}$, that has rules to shuffle constructors, similar to Regnier's $\sigma$-rules for CBN [28], as an alternative to explicit substitutions—it was introduced by Carraro and Guerrieri [8] (and further analyzed in [15,14]) to study the adequacy of Open/Strong CBV with respect to denotational semantics related to linear logic.
4. *The Value Sequent Calculus* $\lambda_{\mathtt{vseq}}$, *i.e.* the intuitionistic fragment of the $\bar{\lambda}\tilde{\mu}$ calculus of Curien and Herbelin [9], that is a CBV calculus for classical logic providing a computational interpretation of sequent calculus rather than natural deduction (in turn a fragment of the $\overline{\lambda}\mu\tilde{\mu}$-calculus [9], further studied in *e.g.* [5,10]).

*Quantitative Analyses and Cost Models.* The number of $\beta_v$-steps is the canonical time cost model of Closed CBV, as first proved by Blelloch and Greiner [7,32,11].

This result is generalized in [1]: the number of steps in $\lambda_{\mathtt{fire}}$ is a reasonable cost model for Open CBV. Here we show that the number of $\beta$-steps in $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{vseq}}$ are *linearly* related to the steps in $\lambda_{\mathtt{fire}}$, thus providing reasonable cost models for these incarnations of Open CBV. As a consequence, complexity analyses can now be smoothly transferred between $\lambda_{\mathtt{fire}}$, $\lambda_{\mathtt{vsub}}$, and $\lambda_{\mathtt{vseq}}$. Said differently, our results guarantee that the number of steps is a *robust* cost model for Open CBV, in the sense that it does not depend on the chosen incarnation of Open CBV. For $\lambda_{\mathtt{sh}}$ we obtain a similar but strictly weaker result, due to some structural difficulties suggesting that $\lambda_{\mathtt{sh}}$ is less apt to complexity analyses.

*On the Value of The Paper.* While the equivalences showed here are new, they might not be terribly surprising. Nonetheless, we think they are interesting, for the following reasons:

1. *Quantitative Relationships*: $\lambda$-calculi are usually related only *qualitatively*, while our relationships are *quantitative* and thus stronger: not only we show simulations, but we also relate the number of steps.
2. *Uniform View*: we provide a new uniform view on a known problem, that will hopefully avoid further proliferations of CBV calculi for open/strong settings.
3. *Expected but Non-Trivial*: while the equivalences are more or less expected, establishing them is informative, because it forces to reformulate and connect concepts among the different settings, and often tricky.
4. *Simple Rewriting Theory*: the relationships between the systems are developed using basic rewriting concepts. The technical development is simple, according to the best tradition of the CBV $\lambda$-calculus, and yet it provides a sharp and detailed decomposition of Open CBV evaluation.
5. *Connecting Different Worlds*: while $\lambda_{\mathtt{fire}}$ is related to Coq and implementations, $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{sh}}$ have a linear logic background, and $\lambda_{\mathtt{vseq}}$ is rooted in sequent calculus. With respect to linear logic, $\lambda_{\mathtt{vsub}}$ has been used for syntactical studies while $\lambda_{\mathtt{sh}}$ for semantical ones. Our results therefore establish bridges between these different (sub)communities.

Finally, an essential contribution of this work is the recognition of Open CBV as a simple and yet rich framework in between Closed and Strong CBV.

*Road map.* Sect. 2 provides an overview of the different presentations of Open CBV. Sect. 3 proves the termination equivalences for $\lambda_{\mathtt{vsub}}$, $\lambda_{\mathtt{fire}}$ and $\lambda_{\mathtt{sh}}$, enriched with quantitative information. Sect. 4 proves, with quantitative information, the termination equivalence for $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{vseq}}$, via an intermediate calculus $\lambda_{\mathtt{vsub}_k}$. Appendix A (p. 19) collects the definitions and notations of the rewriting notions at work in the paper. Omitted proofs are in Appendix B (p. 20). In case of acceptance, this long version with Appendices will be made available on Arxiv.

## 2 Incarnations of Open Call-by-Value

Here we recall Naïve Open CBV $\lambda_{\mathtt{Plot}}$ and introduce the four forms of Open CBV that will be compared ($\lambda_{\mathtt{fire}}$, $\lambda_{\mathtt{vsub}}$, $\lambda_{\mathtt{sh}}$, and $\lambda_{\mathtt{vseq}}$) together with a semantic notion (*potential valuability*) reducing Open CBV to Closed CBV. In this paper

| | |
|---|---|
| Terms | $t, u, s, r ::= v \mid tu$ |
| Values | $v, v' ::= x \mid \lambda x.t$ |
| Evaluation Contexts | $E ::= \langle \cdot \rangle \mid tE \mid Et$ |
| Rule at Top Level | Contextual closure |
| $(\lambda x.t)\lambda y.u \mapsto_{\beta_\lambda} t\{x{\leftarrow}\lambda y.u\}$ | $E\langle t\rangle \to_{\beta_\lambda} E\langle u\rangle$ if $t \mapsto_{\beta_\lambda} u$ |
| $(\lambda x.t)y \mapsto_{\beta_y} t\{x{\leftarrow}y\}$ | $E\langle t\rangle \to_{\beta_y} E\langle u\rangle$ if $t \mapsto_{\beta_y} u$ |
| Reduction | $\to_{\beta_v} := \to_{\beta_\lambda} \cup \to_{\beta_y}$ |

**Fig. 1.** Naïve Open CBV $\lambda_{\mathtt{Plot}}$

terms are always possibly open. Moreover, we focus on Open CBV and avoid on purpose to study Strong CBV (we hint at how to define it, though).

*Naïve Open CBV: Plotkin's calculus* $\lambda_{\mathtt{Plot}}$ [27] Naïve Open CBV is Plotkin's weak CBV $\lambda$-calculus $\lambda_{\mathtt{Plot}}$ on possibly open terms, defined in Fig. 1. Our presentation of the rewriting is unorthodox because we split $\beta_v$-reduction into two rules, according to the kind of value (abstraction or variable). The set of terms is denoted by $\Lambda$. Terms (in $\Lambda$) are always identified up to $\alpha$-equivalence and the set of the free variables of a term $t$ is denoted by $\mathtt{fv}(t)$. We use $t\{x{\leftarrow}u\}$ for the term obtained by the capture-avoiding substitution of $u$ for each free occurrence of $x$ in $t$. Evaluation $\to_{\beta_v}$ is weak and non-deterministic, since in the case of an application there is no fixed order in the evaluation of the left and right subterms. As it is well-known, non-determinism is only apparent: the system is strongly confluent (see Appendix A for a glossary and notations of rewriting theory).

**Proposition 1.** *$\to_{\beta_y}$, $\to_{\beta_\lambda}$ and $\to_{\beta_v}$ are strongly confluent.* Proof p. 20

Strong confluence is a remarkable property, much stronger than plain confluence. It implies that, given a term, all derivations to its normal form (if any) have the same length, and that normalization and strong normalization coincide, *i.e.* if there is a normalizing derivation then there are no diverging derivations. Strong confluence will also hold for $\lambda_{\mathtt{fire}}$, $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{vseq}}$, not for $\lambda_{\mathtt{sh}}$.

Let us come back to the splitting of $\to_{\beta_v}$. In Closed CBV it is well-known that $\to_{\beta_y}$ is superfluous, at least as long as small-step evaluation is considered, see [4]. For Open CBV, $\to_{\beta_y}$ is instead necessary, but—as we explained in Sect. 1—it is not enough, which is why we shall consider extensions of $\lambda_{\mathtt{Plot}}$. The main problem of Naïve Open CBV is that there are stuck $\beta$-redexes that break the harmony of the system. There are three kinds of solution: those *restoring a form of harmony* ($\lambda_{\mathtt{fire}}$), to be thought as more semantical approaches; those *removing stuck $\beta$-redexes* ($\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{sh}}$), that are more syntactical in nature; those *changing the applicative structure of $\lambda$-terms* ($\lambda_{\mathtt{vseq}}$), inspired by sequent calculus.

*Open Call-by-Value 1: The Fireball Calculus* $\lambda_{\mathtt{fire}}$. The Fireball Calculus $\lambda_{\mathtt{fire}}$, defined in Fig. 2, was introduced without a name by Paolini and Ronchi Della Rocca in [26] and [29, Def. 3.1.4, p. 36] where its basic properties are also proved. We give here a presentation inspired by Accattoli and Sacerdoti Coen's [1],

| | |
|---|---|
| Terms and Values | As in Plotkin's Open CBV (Fig. 1) |
| Fireballs | $f, f', f'' ::= \lambda x.t \mid i$ |
| Inert Terms | $i, i', i'' ::= x f_1 \dots f_n \quad n \geq 0$ |
| Evaluation Contexts | $E ::= \langle\cdot\rangle \mid tE \mid Et$ |
| Rule at Top Level | Contextual closure |
| $(\lambda x.t)(\lambda y.u) \mapsto_{\beta_\lambda} t\{x{\leftarrow}\lambda y.u\}$ | $E\langle t\rangle \to_{\beta_\lambda} E\langle u\rangle$ if $t \mapsto_{\beta_\lambda} u$ |
| $(\lambda x.t)i \mapsto_{\beta_i} t\{x{\leftarrow}i\}$ | $E\langle t\rangle \to_{\beta_i} E\langle u\rangle$ if $t \mapsto_{\beta_i} u$ |
| Reduction | $\to_{\beta_f} := \to_{\beta_\lambda} \cup \to_{\beta_i}$ |

**Fig. 2.** The Fireball Calculus $\lambda_{\texttt{fire}}$

departing from it only for inessential, cosmetic details. Terms and evaluation contexts are the same as in $\lambda_{\texttt{Plot}}$.

The idea is to restore harmony by generalizing $\to_{\beta_y}$ to fire when the argument is a more general *inert term*—the new rule is noted $\to_{\beta_i}$. The generalization of values as to include inert terms is called *fireballs*. Actually fireballs and inert terms are defined by mutual recursion (in Fig. 2). For instance, $\lambda x.y$ is a fireball as an abstraction, while $x$, $y(\lambda x.x)$, $xy$, and $(z(\lambda x.x))(zz)(\lambda y.(zy))$ are fireballs as inert terms. Note that $ii'$ is an inert term for all inert terms $i$ and $i'$. Inert terms can be equivalently defined as $i ::= x \mid if$—such a definition is used in the proofs in the Appendix. Inert terms that are not variables are referred to as *compound inert terms*. The main feature of an inert term is that it is normal and that when plugged in a context it cannot create a redex, hence the name (it is not a so-called *neutral term* because it might have redexes under abstractions). In Grégoire and Leroy's presentation [13], inert terms are called *accumulators* and fireballs are simply called values.

Evaluation is given by the fireball rule $\to_{\beta_f}$, that is the union of $\to_{\beta_\lambda}$ and $\to_{\beta_i}$. For instance, consider $t := ((\lambda y.\delta)(zz))\delta$ and $u := \delta((\lambda y.\delta)(zz))$ as in Eq. (1), p. 2: $t$ and $u$ are $\beta_v$-normal but they diverge when evaluated in $\lambda_{\texttt{fire}}$, as desired: $t \to_{\beta_i} \delta\delta \to_{\beta_\lambda} \delta\delta \to_{\beta_\lambda} \dots$ and $u \to_{\beta_i} \delta\delta \to_{\beta_\lambda} \delta\delta \to_{\beta_\lambda} \dots$ .

The distinguished, key property of $\lambda_{\texttt{fire}}$ is (for any $t \in \Lambda$):

Proof p. 21

**Proposition 2 (Open Harmony).** *$t$ is $\beta_f$-normal iff $t$ is a fireball.*

The advantage of $\lambda_{\texttt{fire}}$ is its simple notion of normal form, *i.e.* fireballs, that have a clean syntactic description akin to that for call-by-name. The other calculi will lack a nice, natural notion of normal form. The concepts of $\lambda_{\texttt{fire}}$, however, will allow us to somewhat identify a good notion of normal form also for $\lambda_{\texttt{vsub}}$ and $\lambda_{\texttt{vseq}}$. The drawback of the fireball calculus—and probably the reason why its importance did not emerge before—is the fact that as a strong calculus it is not confluent: this is due to the fact that fireballs are not closed by substitution (see [29, p. 37]). Indeed, if evaluation is strong, the following critical pair cannot be joined, where $t := (\lambda y.I)(\delta\delta)$ and $I := \lambda z.z$ is the identity combinator:

$$I \;{}_{\beta_\lambda}\!\!\leftarrow (\lambda x.I)\delta \;{}_{\beta_i}\!\!\leftarrow (\lambda x.(\lambda y.I)(xx))\delta \to_{\beta_\lambda} t \to_{\beta_\lambda} t \to_{\beta_\lambda} \dots \qquad (2)$$

On the other hand, as long as evaluation is weak (that is the case we consider) everything works fine—the strong case can then be caught by iterating the weak one. In fact, fireball evaluation has a simple rewriting theory, as the next proposition shows. In particular it is strongly confluent.

**Proposition 3 (Basic properties of $\lambda_{\mathtt{fire}}$).** Proof p. 21

1. *$\to_{\beta_i}$ is strongly normalizing and strongly confluent.*
2. *$\to_{\beta_\lambda}$ and $\to_{\beta_i}$ strongly commute.*
3. *$\to_{\beta_f}$ is strongly confluent, and all $\beta_f$-normalizing derivations $d$ from $t \in \Lambda$ (if any) have the same length $|d|_{\beta_f}$, the same number $|d|_{\beta_\lambda}$ of $\beta_\lambda$-steps, and the same number $|d|_{\beta_i}$ of $\beta_i$-steps.*

*Rewriting Interlude: Creations of Type 1 and 4.* The problem with stuck normal forms can be easily understood at the rewriting level as an issue about creations. According to Lévy [20], in the ordinary CBN $\lambda$-calculus redexes can be created in 3 ways. Creations of type 1 take the following form

$$((\lambda x.\lambda y.t)r)s \to_\beta (\lambda y.t\{x{\leftarrow}r\})s$$

where the redex involving $\lambda y$ and $s$ has been created by the $\beta$-step. In Naïve Open CBV if $r$ is a normal form and not a value then the creation cannot take place, blocking evaluation. This is the problem concerning the term $t$ in Eq. (1), p. 2. In CBV there is another form of creation—of *type 4*—not considered by Lévy:

$$(\lambda x.t)((\lambda y.v)v') \to_{\beta_v} (\lambda x.t)(v\{y{\leftarrow}v'\})$$

*i.e.* a reduction in the argument turns the argument itself into a value, creating a $\beta_v$-redex. As before, in a open setting $v'$ may be replaced by a normal form that is not a value, blocking the creation of type 4. This is exactly the problem concerning the term $u$ in Eq. (1), p. 2.

The next two proposals for Open CBV essentially introduce some way to enable creations of type 1 and 4, without substituting stuck $\beta$-redexes nor inert terms.

*Open Call-by-Value 2: The Value Substitution Calculus $\lambda_{\mathtt{vsub}}$.* The *value substitution calculus* $\lambda_{\mathtt{vsub}}$ of Accattoli and Paolini [3,2] was introduced as a calculus for Strong CBV inspired by linear logic proof nets. In Fig. 3 we present its adaptation to Open CBV, obtained by simply removing abstractions from evaluation contexts. It extends the syntax of terms with the constructor $[x{\leftarrow}u]$, called *explicit substitution* (shortened ES, to not be confused with the meta-level substitution $\{x{\leftarrow}u\}$). A `vsub`-term $t[x{\leftarrow}u]$ represents the delayed substitution of $u$ for $x$ in $t$, *i.e.* stands for `let` $x = u$ `in` $t$. So, $t[x{\leftarrow}u]$ binds the free occurrences of $x$ in $t$. The set of `vsub`-terms—identified up to $\alpha$-equivalence—is denoted by $\Lambda_{\mathtt{vsub}}$ (clearly $\Lambda \subsetneq \Lambda_{\mathtt{vsub}}$).

ES are used to remove stuck $\beta$-redexes: the idea is that $\beta$-redexes can be fired whenever—even if the argument is not a (`vsub`-)value—by means of the *multiplicative rule* $\to_{\mathtt{m}}$; however the argument is not substituted but placed in a ES. The actual substitution is done only when the content of the ES is a `vsub`-value, by means of the *exponential rule* $\to_{\mathtt{e}}$. These two rules are sometimes noted $\to_{\mathtt{dB}}$ ($\beta$ at a distance) and $\to_{\mathtt{vs}}$ (substitution by value)—the names we use here

| | |
|---|---|
| vsub-Terms | $t, u, s ::= v \mid tu \mid t[x{\leftarrow}u]$ |
| vsub-Values | $v ::= x \mid \lambda x.t$ |
| Evaluation Contexts | $E ::= \langle\cdot\rangle \mid tE \mid Et \mid E[x{\leftarrow}u] \mid t[x{\leftarrow}E]$ |
| Substitution Contexts | $L ::= \langle\cdot\rangle \mid L[x{\leftarrow}u]$ |

| Rule at Top Level | Contextual closure |
|---|---|
| $L\langle\lambda x.t\rangle u \mapsto_{\mathsf{m}} L\langle t[x{\leftarrow}u]\rangle$ | $E\langle t\rangle \to_{\mathsf{m}} E\langle u\rangle$ if $t \mapsto_{\mathsf{m}} u$ |
| $t[x{\leftarrow}L\langle\lambda y.u\rangle] \mapsto_{\mathsf{e}_\lambda} L\langle t\{x{\leftarrow}\lambda y.u\}\rangle$ | $E\langle t\rangle \to_{\mathsf{e}_\lambda} E\langle u\rangle$ if $t \mapsto_{\mathsf{e}_\lambda} u$ |
| $t[x{\leftarrow}L\langle y\rangle] \mapsto_{\mathsf{e}_y} L\langle t\{x{\leftarrow}y\}\rangle$ | $E\langle t\rangle \to_{\mathsf{e}_y} E\langle u\rangle$ if $t \mapsto_{\mathsf{e}_y} u$ |

Reductions $\quad \to_{\mathsf{e}} := \to_{\mathsf{e}_\lambda} \cup \to_{\mathsf{e}_y}, \quad \to_{\mathtt{vsub}} := \to_{\mathsf{m}} \cup \to_{\mathsf{e}}$

**Fig. 3.** The Value Substitution Calculus $\lambda_{\mathtt{vsub}}$

are due to the interpretation of the calculus into linear logic proof-nets, see [2]. A characteristic feature coming from such an interpretation is that the rewriting rules are contextual, or *at a distance*: they are generalized as to act up to a list of substitutions (noted $L$, from $L$ist). Essentially, stuck $\beta$-redexes are turned into ES and then ignored by the rewriting rules—this is how creations of type 1 and 4 are enabled. For instance, the terms $t := ((\lambda y.\delta)(zz))\delta$ and $u := \delta((\lambda y.\delta)(zz))$ (as in Eq. (1), p. 2) are $\mathsf{e}$-normal but $t \to_{\mathsf{m}} \delta[y{\leftarrow}zz]\delta \to_{\mathsf{m}} (xx)[x{\leftarrow}\delta][y{\leftarrow}zz] \to_{\mathsf{e}} (\delta\delta)[y{\leftarrow}zz] \to_{\mathsf{m}} (xx)[x{\leftarrow}\delta][y{\leftarrow}zz] \to_{\mathsf{e}} (\delta\delta)[y{\leftarrow}zz] \to_{\mathsf{m}} \dots$ and similarly for $u$.

The drawback of $\lambda_{\mathtt{vsub}}$ is that it requires explicit substitutions. The advantage of $\lambda_{\mathtt{vsub}}$ is its simple and well-behaved rewriting theory, even simpler than the rewriting for $\lambda_{\mathtt{fire}}$, as every rule terminates separately (while $\beta_\lambda$ does not)—in particular strong confluence holds. Moreover, the theory has a sort of flexible second level given by a notion of structural equivalence, coming up next.

Proof p. 23

**Proposition 4 (Basic Properties of $\lambda_{\mathtt{vsub}}$, [3]).**

1. $\to_{\mathsf{m}}$ *and* $\to_{\mathsf{e}}$ *are strongly normalizing and strongly confluent (separately).*
2. $\to_{\mathsf{m}}$ *and* $\to_{\mathsf{e}}$ *strongly commute.*
3. $\to_{\mathtt{vsub}}$ *is strongly confluent, and all* vsub*-normalizing derivations* $d$ *from* $t \in \Lambda_{\mathtt{vsub}}$ *(if any) have the same length* $|d|_{\mathtt{vsub}}$*, the same number* $|d|_{\mathsf{e}}$ *of* e*-steps, and the same number* $|d|_{\mathsf{m}}$ *of* m*-steps*
4. *Let* $t \in \Lambda$*. For any* vsub*-derivation* $d$ *from* $t$*,* $|d|_{\mathsf{e}} \leq |d|_{\mathsf{m}}$.

*Structural Equivalence.* The theory of $\lambda_{\mathtt{vsub}}$ comes with a notion of structural equivalence $\equiv$, that equates vsub-terms that differ only for the position of ES. The basic idea is that the action of an ES via the exponential rule depends on the position of the ES itself only for inessential details (as long as the scope of binders is respected), namely the position of other ES, and thus can be abstracted away. A strong justification for the equivalence comes from the linear logic interpretation of the call-by-value $\lambda$-calculus, in which structurally equivalent vsub-terms translate to the same (recursively typed) proof net, see [2].

Structural equivalence $\equiv$ is defined as the least equivalence relation on $\Lambda_{\mathtt{vsub}}$ closed by evaluation contexts (see Fig. 3) and generated by the following axioms:

$$\begin{aligned} t[y{\leftarrow}s][x{\leftarrow}u] &\equiv_{\text{com}} t[x{\leftarrow}u][y{\leftarrow}s] && \text{if } y \notin \mathtt{fv}(u) \text{ and } x \notin \mathtt{fv}(s) \\ t\,s[x{\leftarrow}u] &\equiv_{@\text{r}} (ts)[x{\leftarrow}u] && \text{if } x \notin \mathtt{fv}(t) \\ t[x{\leftarrow}u]s &\equiv_{@\text{l}} (ts)[x{\leftarrow}u] && \text{if } x \notin \mathtt{fv}(s) \\ t[x{\leftarrow}u[y{\leftarrow}s]] &\equiv_{[\cdot]} t[x{\leftarrow}u][y{\leftarrow}s] && \text{if } y \notin \mathtt{fv}(t) \end{aligned}$$

We set $\to_{\mathtt{vsub}_\equiv} := \equiv\to_{\mathtt{vsub}}\equiv$ (*i.e.* for all $t, r \in \Lambda_{\mathtt{vsub}}$: $t \to_{\mathtt{vsub}_\equiv} r$ iff $t \equiv u \to_{\mathtt{vsub}} s \equiv r$ for some $u, s \in \Lambda_{\mathtt{vsub}}$). The notation $\to^+_{\mathtt{vsub}_\equiv}$ keeps its usual meaning, while $\to^*_{\mathtt{vsub}_\equiv}$ stands for $\equiv \cup \to^+_{\mathtt{vsub}_\equiv}$, *i.e.* a $\mathtt{vsub}_\equiv$-derivation of length zero can apply $\equiv$ and is not just the identity. As $\equiv$ is reflexive, $\to_{\mathtt{vsub}} \subsetneq \to_{\mathtt{vsub}_\equiv}$.

The rewriting theory of $\lambda_{\mathtt{vsub}}$ enriched with structural equivalence $\equiv$ is remarkably simple, as the next lemma shows. In fact, $\equiv$ commutes with evaluation, and can thus be postponed. Additionally, the commutation is *strong*, as it preserves the number and kind of steps—one says that it is a *strong bisimulation* (with respect to $\to_{\mathtt{vsub}}$). In particular, the equivalence is not needed to compute and it does not break, or make more complex, any property of $\lambda_{\mathtt{vsub}}$. On the contrary, it enhances the flexibility of the system: it will be essential to establish simple and clean relationships with the other calculi for Open CBV.

**Lemma 5 (Basic Properties of Structural Equivalence $\equiv$, [3]).** *Let $t, u \in \Lambda_{\mathtt{vsub}}$ and $\mathsf{x} \in \{\mathsf{m}, \mathsf{e}_\lambda\mathsf{e}_y, \mathsf{e}, \mathtt{vsub}\}$.*

1. Strong Bisimulation of $\equiv$ wrt $\to_{\mathtt{vsub}}$*: if $t \equiv u$ and $t \to_{\mathsf{x}} t'$ then there exists $u' \in \Lambda_{\mathtt{vsub}}$ such that $u \to_{\mathsf{x}} u'$ and $t' \equiv u'$.*
2. Postponement of $\equiv$ wrt $\to_{\mathtt{vsub}}$*: if $d\colon t \to^*_{\mathtt{vsub}_\equiv} u$ then there are $s \equiv u$ and $e\colon t \to^*_{\mathtt{vsub}} s$ such that $|d| = |e|$, $|d|_{\mathsf{e}_\lambda} = |e|_{\mathsf{e}_\lambda}$, $|d|_{\mathsf{e}_y} = |e|_{\mathsf{e}_y}$ and $|d|_{\mathsf{m}} = |e|_{\mathsf{m}}$.*
3. Normal Forms*: if $t \equiv u$ then $t$ is $\mathsf{x}$-normal iff $u$ is $\mathsf{x}$-normal.*
4. Strong confluence*: $\to_{\mathtt{vsub}_\equiv}$ is strongly confluent.*

*Open Call-by-Value 3: The Shuffling Calculus $\lambda_{\mathtt{sh}}$.* The calculus introduced by Carraro and Guerrieri in [8], and here deemed *Shuffling Calculus*, has the same syntax of terms as Plotkin's calculus. Two additional commutation rules help $\to_{\beta_v}$ to deal with stuck $\beta$-redexes, by shuffling constructors so as to enable creations of type 1 and 4. As for $\lambda_{\mathtt{vsub}}$, $\lambda_{\mathtt{sh}}$ was actually introduced, and then used in [8,14,15], to study Strong CBV. In Fig. 4 we present its adaptation to Open CBV, based on *balanced contexts*, a special notion of evaluation contexts. The reductions $\to_{\sigma^\flat}$ and $\to_{\beta_v^\flat}$ are non-deterministic and—because of balanced contexts—can reduce under abstractions, but they are *morally* weak: they reduce under a $\lambda$ only when the $\lambda$ is applied to an argument. Note that the condition $x \notin \mathtt{fv}(s)$ (resp. $x \notin \mathtt{fv}(v)$) in the definition of the shuffling rule $\mapsto_{\sigma_1}$ (resp. $\mapsto_{\sigma_3}$) can always be fulfilled by $\alpha$-conversion.

The reduction $\to_{\sigma^\flat}$ unblocks stuck $\beta$-redexes. For instance, consider the terms $t := ((\lambda y.\delta)(zz))\delta$ and $u := \delta((\lambda y.\delta)(zz))$ where $\delta := \lambda x.xx$ (as in Eq. (1), p. 2): $t$ and $u$ are $\beta_v^\flat$-normal but $t \to_{\sigma_1^\flat} (\lambda y.\delta\delta)(zz) \to_{\beta_v^\flat} (\lambda y.\delta\delta)(zz) \to_{\beta_v^\flat} \ldots$ and $u \to_{\sigma_3^\flat} (\lambda y.\delta\delta)(zz) \to_{\beta_v^\flat} (\lambda x.\delta\delta)(zz) \to_{\beta_v^\flat} \ldots$ .

| | |
|---|---|
| Terms and Values | As in Plotkin's Open CBV (Fig. 1) |
| Balanced Contexts | $B ::= \langle\cdot\rangle \mid tB \mid Bt \mid (\lambda x.B)t$ |

| Rule at Top Level | Contextual closure |
|---|---|
| $((\lambda x.t)u)s \mapsto_{\sigma_1} (\lambda x.ts)u,\ x \notin \mathtt{fv}(s)$ | $B\langle t\rangle \to_{\sigma_1^\flat} B\langle u\rangle$ if $t \mapsto_{\sigma_1} u$ |
| $v((\lambda x.s)u) \mapsto_{\sigma_3} (\lambda x.vs)u,\ x \notin \mathtt{fv}(v)$ | $B\langle t\rangle \to_{\sigma_3^\flat} B\langle u\rangle$ if $t \mapsto_{\sigma_3} u$ |
| $(\lambda x.t)v \mapsto_{\beta_v} t\{x{\leftarrow}v\}$ | $B\langle t\rangle \to_{\beta_v^\flat} B\langle u\rangle$ if $t \mapsto_{\beta_v} u$ |

Reductions $\quad \to_{\sigma^\flat} := \to_{\sigma_1^\flat} \cup \to_{\sigma_3^\flat}, \quad \to_{\mathtt{sh}} := \to_{\beta_v^\flat} \cup \to_{\sigma^\flat}$

**Fig. 4.** Shuffling $\lambda$-calculus $\lambda_{\mathtt{sh}}$

The similar shuffling rules in CBN, better known as Regnier's $\sigma$-rules [28], are *contained* in CBN $\beta$-equivalence, while in Open (and Strong) CBV they are more interesting, as they are not contained into (*i.e.* they enrich) $\beta_v$-equivalence.

The advantage of $\lambda_{\mathtt{sh}}$ is with respect to denotational investigations. In [8], $\lambda_{\mathtt{sh}}$ is indeed used to prove various semantical results in connection to linear logic, resource calculi, and the notion of Taylor expansion due to Ehrhard. In particular, in [8] it has been proved the adequacy of $\lambda_{\mathtt{sh}}$ with respect to the relational model induced by linear logic: a by-product of our paper is the extension of this adequacy result to all incarnations of Open CBV. The drawback of $\lambda_{\mathtt{sh}}$ is its technical rewriting theory. We summarize some properties of $\lambda_{\mathtt{sh}}$:

Proof p. 27

**Proposition 6 (Basic Properties of $\lambda_{\mathtt{sh}}$, [8]).**
1. *Let $t, u, s \in \Lambda$. If $t \to_{\beta_v^\flat} u$ and $t \to_{\sigma^\flat} s$ then $u \neq s$.*
2. *$\to_{\sigma^\flat}$ is strongly normalizing and (not strongly) confluent.*
3. *$\to_{\mathtt{sh}}$ is (not strongly) confluent.*
4. *Let $t \in \Lambda$: $t$ is strongly* sh*-normalizable iff $t$ is* sh*-normalizable.*

In contrast to $\lambda_{\mathtt{fire}}$ and $\lambda_{\mathtt{vsub}}$, $\lambda_{\mathtt{sh}}$ is not strongly confluent and not all sh-normalizing derivations (if any) from a given term have the same length (consider, for instance, all sh-normalizing derivations from $(\lambda y.z)(\delta(zz))\delta)$. Nonetheless, normalization and strong normalization still coincide (Prop. 6.4), and Cor. 18 in Sect. 3 will show that the discrepancy is encapsulated inside the additional shuffling rules, as all sh-normalizing derivations from a given term have the same number of $\beta_v^\flat$-steps.

*Open Call-by-Value 4: The Value Sequent Calculus* $\lambda_{\mathtt{vseq}}$. A more radical approach to the removal of stuck $\beta$-redexes is provided by what here we call the *Value Sequent Calculus* $\lambda_{\mathtt{vseq}}$, defined in Fig. 5, where it is the applicative structure of terms that is altered, by replacing the application constructor with more constructs, namely commands $c$ and environments $e$. Morally, $\lambda_{\mathtt{vseq}}$ looks at a sequence of applications from the head, that is the value on the left of a command $\langle v \,|\, e\rangle$ rather than from the tail as in natural deduction. In fact, $\lambda_{\mathtt{vseq}}$ is a handy presentation of the intuitionistic fragment of $\bar{\lambda}\tilde{\mu}$, that in turn is the CBV fragment of $\bar{\lambda}\mu\tilde{\mu}$, a calculus obtained as the computational interpretation of a sequent calculus for classical logic. Both $\bar{\lambda}\tilde{\mu}$ and $\bar{\lambda}\mu\tilde{\mu}$ are due to Curien and Herbelin [9], see *e.g.* [5,10] for further investigations about these systems.

| | |
|---|---|
| Commands | $c, c' ::= \langle v \,|\, e \rangle$ |
| Values | $v, v' ::= x \mid \lambda x.c$ |
| Environments | $e, e' ::= \epsilon \mid \tilde{\mu}x.c \mid v \cdot e$ |
| Command Evaluation Contexts | $C ::= \langle \cdot \rangle \mid D\langle \tilde{\mu}x.C \rangle$ |
| Environment Evaluation Contexts | $D ::= \langle v \,|\, \langle \cdot \rangle \rangle \mid D\langle v \cdot \langle \cdot \rangle \rangle$ |
| Rule at Top Level | Contextual closure |
| $\langle \lambda x.c \,|\, v \cdot e \rangle \mapsto_{\bar{\lambda}} \langle v \,|\, (\tilde{\mu}x.c)@e \rangle$ | $C\langle c \rangle \to_{\bar{\lambda}} C\langle c' \rangle$ if $c \mapsto_{\bar{\lambda}} c'$ |
| $\langle v \,|\, \tilde{\mu}x.c \rangle \mapsto_{\tilde{\mu}} c\{x \leftarrow v\}$ | $C\langle c \rangle \to_{\tilde{\mu}} C\langle c' \rangle$ if $c \mapsto_{\tilde{\mu}} c'$ |
| Reduction | $\to_{\mathtt{vseq}} := \to_{\bar{\lambda}} \cup \to_{\tilde{\mu}}$ |

**Fig. 5.** Open CBV 4: the $\lambda_{\mathtt{vseq}}$-Calculus

A peculiar trait of the sequent calculus approach is the environment constructor $\tilde{\mu}x.c$, that is a binder for the free occurrences of $x$ in $c$. It is often said that it is a sort of explicit substitution—we will see exactly in which sense, in Sect. 4.

The change of the intuitionistic variant $\lambda_{\mathtt{vseq}}$ with respect to $\bar{\lambda}\tilde{\mu}$ is that $\lambda_{\mathtt{vseq}}$ does not need the syntactic category of co-variables $\alpha$, as there can be only one of them, that we note $\epsilon$. From a logic point of view, this is due to the fact that in intuitionistic sequent calculus, there is neither contraction nor weakening on the right-hand-side of $\vdash$. Consequently, the binary abstraction $\lambda(x, \alpha).c$ of $\bar{\lambda}\tilde{\mu}$ is replaced by a more traditional unary one $\lambda x.c$, and substitution on co-variables is replaced by a notion of *appending of environments*, defined by mutual induction on commands and environments as follows:

$$\langle v \,|\, e' \rangle @e := \langle v \,|\, e'@e \rangle \qquad \epsilon@e := e$$
$$(v \cdot e')@e := v \cdot (e'@e) \quad (\tilde{\mu}x.c)@e := \tilde{\mu}y.(c\{x \leftarrow y\}@e) \text{ with } y \notin \mathtt{fv}(c) \cup \mathtt{fv}(e)$$

Essentially, $c@e$ is a capture-avoiding substitution of $e$ for the only occurrence of $\epsilon$ in $c$ that is out of all abstractions, which stands for the output of the term. The append operation is used in $\to_{\bar{\lambda}}$, one of the two rewrite rules of $\lambda_{\mathtt{vseq}}$ (Fig. 5). Strong CBV can be obtained by simply extending the grammar of evaluation contexts to commands under abstractions.

We will provide a translation from $\lambda_{\mathtt{vsub}}$ to $\lambda_{\mathtt{vseq}}$, that beyond the termination equivalence, will show that the switching to the sequent calculus representation is equivalent to a transformation in administrative normal form [30].

The advantage of $\lambda_{\mathtt{vseq}}$ is that it avoids both rules at a distance and shuffling rules. The drawback of $\lambda_{\mathtt{vseq}}$ is that, syntactically, it requires to step out of the $\lambda$-calculus. We will show in Sect. 4 how to reformulate it as a fragment of $\lambda_{\mathtt{vsub}}$, *i.e.* in natural deduction. However, it will still be necessary to restrict the application constructor, thus preventing the natural way of writing $\lambda$-terms.

The rewriting of $\lambda_{\mathtt{vseq}}$ is very well-behaved, in particular it is strongly confluent and the rules terminates separately.

**Proposition 7 (Basic properties of $\lambda_{\mathtt{vseq}}$).** Proof p. 30

1. *$\to_{\bar{\lambda}}$ and $\to_{\tilde{\mu}}$ are strongly normalizing and strongly confluent (separately).*
2. *$\to_{\bar{\lambda}}$ and $\to_{\tilde{\mu}}$ strongly commute.*

*3.* $\to_{\mathtt{vseq}}$ *is strongly confluent, and all* vseq*-normalizing derivations $d$ from a command $c$ (if any) have the same length $|d|$, the same number $|d|_{\tilde{\mu}}$ of $\tilde{\mu}$-steps, and the same number $|d|_{\bar{\lambda}}$ of $\bar{\lambda}$-steps.*

*Reducing Open to Closed Call-by-Value: Potential Valuability.* Potential valuability relates Naïve Open CBV to Closed CBV via a meta-level substitution closing open terms: a (possibly open) term $t$ is *potentially valuable* if there is a substitution of (closed) *values* for its free variables, for which it $\beta_v$-evaluates to a (closed) *value*. In Naïve Open CBV, potentially valuable terms do not coincide with normalizable terms because of premature $\beta_v$-normal forms—as $t$ and $u$ in Eq. (1) at p. 2— which are not potentially valuable.

Paolini, Ronchi Della Rocca and, later, Pimentel [26,24,29,25,23] gave several operational, logical, and semantical characterizations of potentially valuable terms in Naïve Open CBV. In particular, in [26,29] it is proved that a term is potentially valuable in Plotkin's Naïve Open CBV iff its normalizable in $\lambda_{\mathtt{fire}}$.

Potentially valuable terms can be defined for every incarnation of Open CBV: it is enough to update the notions of evaluation and values in the above definition to the considered calculus. This has been done for $\lambda_{\mathtt{sh}}$ in [8], and for $\lambda_{\mathtt{vsub}}$ in [3]. For both calculi it has been proved that, in the weak setting, potentially valuable terms coincides with normalizable terms. In [15], it has been proved that Plotkin's potentially valuable terms coincide with $\lambda_{\mathtt{sh}}$-potentially valuable terms (which coincide in turn with sh-normalizable terms). Our paper makes a further step: proving that termination coincides for $\lambda_{\mathtt{fire}}$, $\lambda_{\mathtt{vsub}}$, $\lambda_{\mathtt{sh}}$, and $\lambda_{\mathtt{vseq}}$ it implies that all their notions of potential valuability coincide with Plotkin's, *i.e.* there is just one notion of potential valuability.

*Open CBV 5, 6, 7, ...* The literature contains many other calculi for CBV, usually presented for Strong CBV and easily adaptable to Open CBV. Some of them have `let`-expressions (avatars of ES) and all of them have rules permuting constructors, therefore they lie somewhere in between $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{sh}}$. Often, they have been developed for other purposes, usually to investigate the relationship with monad or CPS translations. Moggi's equational theory [22] is a classic standard of reference, known to coincide with that of Sabry and Felleisen [30], Sabry and Wadler [31], Dychoff and Lengrand [12], Herbelin and Zimmerman [17] and Maraist et al's $\lambda_{let}$ in [21]. In [3], $\lambda_{\mathtt{vsub}}$ modulo $\equiv$ is shown to be termination equivalent to Herbelin and Zimmerman's calculus, and to strictly contain its equational theory, and thus Moggi's. At the level of rewriting these presentations of Open CBV are all more involved than those that we consider here. Their equivalence to our calculi can be shown along the lines of that of $\lambda_{\mathtt{sh}}$ with $\lambda_{\mathtt{vsub}}$.

## 3 Quantitative Equivalence of $\lambda_{\mathtt{fire}}$, $\lambda_{\mathtt{vsub}}$, and $\lambda_{\mathtt{sh}}$

Here we show the equivalence with respect to termination of $\lambda_{\mathtt{fire}}$, $\lambda_{\mathtt{vsub}}$, and $\lambda_{\mathtt{sh}}$, enriched with quantitative information on the number of steps. The results are obtained simulating both $\lambda_{\mathtt{fire}}$ and $\lambda_{\mathtt{sh}}$ into $\lambda_{\mathtt{vsub}}$. In both cases, structural equivalence $\equiv$ of $\lambda_{\mathtt{vsub}}$ plays a role.

*Simulating* $\lambda_{\texttt{fire}}$ *in* $\lambda_{\texttt{vsub}}$. A single $\beta_v$-step $(\lambda x.t)v \to_{\beta_v} t\{x{\leftarrow}v\}$ is simulated in $\lambda_{\texttt{vsub}}$ by two steps (Lemma 8.1): $(\lambda x.t)v \to_{\mathtt{m}} t[x{\leftarrow}v] \to_{\mathtt{e}} t\{x{\leftarrow}v\}$, *i.e.* a $\mathtt{m}$-step that creates a ES, and a $\mathtt{e}$-step that turns the ES into the meta-level substitution performed by the $\beta_v$-step. The simulation of an inert step of $\lambda_{\texttt{fire}}$ is instead trickier, because in $\lambda_{\texttt{vsub}}$ there is no rule to substitute an inert term, if it is not a variable. The idea is that an inert step $(\lambda x.t)i \to_{\beta_i} t\{x{\leftarrow}i\}$ is simulated only by $(\lambda x.t)i \to_{\mathtt{m}} t[x{\leftarrow}i]$, *i.e.* only by the $\mathtt{m}$-step that creates the ES, and such a ES will never be fired—so the simulation is up to the unfolding of substitutions containing inert terms (defined right next). Everything works because of the key property of inert terms: they are normal and their substitution cannot create redexes, so it is useless to substitute them.

The *unfolding* of a $\texttt{vsub}$-term $t$ is the term $t{\downarrow}$ obtained from $t$ by turning ES into meta-level substitutions; it is defined by:

$$x{\downarrow} := x \qquad (tu){\downarrow} := t{\downarrow}\,u{\downarrow} \qquad (\lambda x.t){\downarrow} := \lambda x.t{\downarrow} \qquad (t[x{\leftarrow}u]){\downarrow} := t{\downarrow}\{x{\leftarrow}u{\downarrow}\}$$

For all $t, u \in \Lambda_{\texttt{vsub}}$, $t \equiv u$ implies $t{\downarrow} = u{\downarrow}$. Also, $t{\downarrow} = t$ iff $t \in \Lambda$.

In the simulation we are going to show, structural equivalence $\equiv$ plays a role. It is used to *clean* the $\texttt{vsub}$-terms (with ES) obtained by simulation, putting them in a canonical form where ES do not appear among other constructors.

A $\texttt{vsub}$-term is *clean* if it has the form $u[x_1{\leftarrow}i_1]\dots[x_n{\leftarrow}i_n]$ (with $n \in \mathbb{N}$), $u \in \Lambda$ is called the *body*, and $i_1, \dots, i_n \in \Lambda$ are inert terms. Clearly, any term (as it is without ES) is clean. We first show how to simulate a single fireball step.

**Lemma 8 (Simulation of a $\beta_f$-Step in $\lambda_{\texttt{vsub}}$).** *Let $t, u \in \Lambda$.* Proof p. 33

1. *If $t \to_{\beta_\lambda} u$ then $t \to_{\mathtt{m}}\to_{\mathtt{e}_\lambda} u$.*
2. *If $t \to_{\beta_i} u$ then $t \to_{\mathtt{m}}\equiv s$, with $s \in \Lambda_{\texttt{vsub}}$ clean and $s{\downarrow} = u$.*

Unfortunately, it is not possible to simulate derivations by iterating Lemma 8, because the starting term $t$ has no ES but the simulation of inert steps introduces ES. Therefore, we have to generalize the statement up to the unfolding of ES. In general, unfolding ES is a dangerous operation with respect to (non-)termination, as it may erase a diverging subterm (*e.g.* $t := x[y{\leftarrow}\delta\delta]$ is $\texttt{vsub}$-divergent and $t{\downarrow} = x$ is normal). In our case, however, the simulation produces clean $\texttt{vsub}$-terms, and so the unfolding is safe because it can only erase inert terms, that cannot create, erase, nor carry redexes.

By means of a technical lemma in the appendix we obtain:

**Lemma 9 (Projection of a $\beta_f$-Step on $\to_{\texttt{vsub}}$ via Unfolding).** *Let $t$ be a clean $\texttt{vsub}$-term and $u$ be a term.* Proof p. 34

1. *If $t{\downarrow} \to_{\beta_\lambda} u$ then $t \to_{\mathtt{m}}\to_{\mathtt{e}_\lambda} s$, with $s \in \Lambda_{\texttt{vsub}}$ clean and $s{\downarrow} = u$.*
2. *If $t{\downarrow} \to_{\beta_i} u$ then $t \to_{\mathtt{m}}\equiv s$, with $s \in \Lambda_{\texttt{vsub}}$ clean and $s{\downarrow} = u$.*

Via Lemma 9 we can now simulate whole derivations. To obtain the termination equivalence, however, we have to work a little bit more. First of all, let us characterize the terms in $\lambda_{\texttt{vsub}}$ obtained by projecting normalizing derivations (that always produce a fireball).

**Lemma 10.** *Let $t$ be a clean* $\mathtt{vsub}$*-term. If $t{\downarrow}$ is a fireball, then $t$ is $\{\mathtt{m}, \mathtt{e}_\lambda\}$-normal and its body is a fireball.* Proof p. 35

Now, a $\{\mathtt{m}, \mathtt{e}_\lambda\}$-normal form $t$ morally is $\mathtt{vsub}$-normal, as $\to_{\mathtt{e}_y}$ terminates (Prop. 4.1) and it cannot create $\{\mathtt{m}, \mathtt{e}_\lambda\}$-redexes. The part about creations is better expressed as a postponement property.

Proof p. 35

**Lemma 11 (Linear Postponement of $\to_{\mathtt{e}_y}$).** *Let $t, u \in \Lambda_{\mathtt{vsub}}$. If $d\colon t \to^*_{\mathtt{vsub}} u$ then $e\colon t \to^*_{\mathtt{m},\mathtt{e}_\lambda} \to^*_{\mathtt{e}_y} u$ with $|e|_{\mathtt{vsub}} = |d|_{\mathtt{vsub}}$, $|e|_{\mathtt{m}} = |d|_{\mathtt{m}}$, $|e|_{\mathtt{e}} = |d|_{\mathtt{e}}$ and $|e|_{\mathtt{e}_\lambda} \geq |d|_{\mathtt{e}_\lambda}$.*

The next theorem puts all the pieces together.

Proof p. 38

**Theorem 12 (Quantitative Simulation of $\lambda_{\mathtt{fire}}$ in $\lambda_{\mathtt{vsub}}$).** *Let $t, u \in \Lambda$. If $d\colon t \to^*_{\beta_f} u$ then there are $s, r \in \Lambda_{\mathtt{vsub}}$ and $e\colon t \to^*_{\mathtt{vsub}} r$ such that*

1. Qualitative Relationship*: $r \equiv s$, $u = s{\downarrow} = r{\downarrow}$ and $s$ is clean;*
2. Quantitative Relationship*:*
   1. Multiplicative Steps: $|d|_{\beta_f} = |e|_{\mathtt{m}}$*;*
   2. Exponential (Abstraction) Steps: $|d|_{\beta_\lambda} = |e|_{\mathtt{e}_\lambda} = |e|_{\mathtt{e}}$.
3. Normal Forms*: if $u$ is $\beta_f$-normal then there exists $f\colon r \to^*_{\mathtt{e}_y} q$ such that $q$ is a* $\mathtt{vsub}$*-normal form and $|f|_{\mathtt{e}_y} \leq |e|_{\mathtt{m}} - |e|_{\mathtt{e}_\lambda}$.*

Proof p. 38

**Corollary 13 (Linear Termination Equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{fire}}$).** *Let $t \in \Lambda$. There is a $\beta_f$-normalizing derivation $d$ from $t$ iff there is a* $\mathtt{vsub}$*-normalizing derivation $e$ from $t$. Moreover, $|d|_{\beta_f} \leq |e|_{\mathtt{vsub}} \leq 2|d|_{\beta_f}$, i.e. they are linearly related.*

The statement of Cor. 13 is stronger than it may look at first sight, because by strong confluence in both $\lambda_{\mathtt{fire}}$ and $\lambda_{\mathtt{vsub}}$, given a term $t$, if there is a normalizing derivation from $t$ then there are no diverging derivations from $t$, and *all* normalizing derivations from $t$ have the same length (Prop. 3.3 and Prop. 4.3).

Since the number of steps in $\lambda_{\mathtt{fire}}$ is known to be a reasonable cost model for Open CBV [1], our result states that also *the number of steps in $\lambda_{\mathtt{vsub}}$ is a reasonable cost model*, and moreover that $\lambda_{\mathtt{fire}}$ and $\lambda_{\mathtt{vsub}}$ are tightly related. Not only the relationship between the two is linear, but the number of multiplicative steps in $\lambda_{\mathtt{vsub}}$ is *exactly* the number of steps in $\lambda_{\mathtt{fire}}$ (Thm. 12.2). By the way, this is somewhat surprising: in $\lambda_{\mathtt{fire}}$ arguments of $\beta_f$-redexes are required to be fireballs, while for $\mathtt{m}$-redexes there are no restrictions on arguments, and yet in every normalizing derivation from a given term their number coincide.

From Lemma 10 it follows that a clean $\mathtt{vsub}$-normal form is a fireball followed by ES with inert terms. This is a nice description of normal forms for $\lambda_{\mathtt{vsub}}$, inherited from $\lambda_{\mathtt{fire}}$, and a by-product of our study.

*Simulating $\lambda_{\mathtt{sh}}$ in $\lambda_{\mathtt{vsub}}$.* A derivation $d\colon t \to^*_{\mathtt{sh}} u$ in $\lambda_{\mathtt{sh}}$ is simulated via a projection on multiplicative normal forms in $\lambda_{\mathtt{vsub}}$, *i.e.* as a derivation $\mathtt{m}(t) \to^*_{\mathtt{vsub}_\equiv} \mathtt{m}(u)$ (for any $\mathtt{vsub}$-term $t$, its multiplicative and exponential normal forms, denoted by $\mathtt{m}(t)$ and $\mathtt{e}(t)$ respectively, exist and are unique by Prop. 4). Indeed, a $\beta_v^\flat$-step of $\lambda_{\mathtt{sh}}$ is simulated in $\lambda_{\mathtt{vsub}}$ by a $\mathtt{e}$-step followed by some $\mathtt{m}$-steps to reach the m-normal form. Shuffling rules $\to_{\sigma^\flat}$ of $\lambda_{\mathtt{sh}}$ are simulated by the structural equivalence $\equiv$ of $\lambda_{\mathtt{vsub}}$: applying $\mathtt{m}(\cdot)$ to $((\lambda x.t)u)s \to_{\sigma_1^\flat} (\lambda x.(ts))u$ we obtain exactly an instance of the axiom $\equiv_{@l}$ defining $\equiv$: $\mathtt{m}(t)[x{\leftarrow}\mathtt{m}(u)]\mathtt{m}(s) \equiv_{@l} (\mathtt{m}(t)\mathtt{m}(s))[x{\leftarrow}\mathtt{m}(u)]$ (with the side conditions matching exactly). Similarly, $\to_{\sigma_3^\flat}$ projects to $\equiv_{@r}$ or $\equiv_{[\cdot]}$ (depending on whether $v$ in $\to_{\sigma_3^\flat}$ is a variable or an abstraction). Therefore,

**Lemma 14 (Projecting a sh-Step on $\to_{\mathtt{vsub}_\equiv}$ via m-nf).** *Let $t, u \in \Lambda$.* Proof p. 40

1. *If $t \to_{\sigma^\flat} u$ then $\mathtt{m}(t) \equiv \mathtt{m}(u)$.*
2. *If $t \to_{\beta_v^\flat} u$ then $\mathtt{m}(t) \to_{\mathtt{e}}\to_{\mathtt{m}}^* \mathtt{m}(u)$.*

In contrast to the simulation of $\lambda_{\mathtt{fire}}$ in $\lambda_{\mathtt{vsub}}$, here the projection of a single step can be extended to derivations without problems, obtaining that the number of $\beta_v^\flat$-steps in $\lambda_{\mathtt{sh}}$ matches exactly the number of e-steps in $\lambda_{\mathtt{vsub}}$. Additionally, we apply the postponement of $\equiv$ (Lemma 5.2), factoring out the use of $\equiv$ (*i.e.* of shuffling rules) without affecting the number of e-steps. So, via Lemma 14 we can now simulate whole derivations. To obtain the termination equivalence, however, we need the following lemma:

**Lemma 15 (Projection Preserves Normal Forms).** *Let $t \in \Lambda$. If $t$ is* sh-*normal then $\mathtt{m}(t)$ is* vsub-*normal.* Proof p. 40

The next theorem puts all the pieces together (for any sh-derivation $d$, $|d|_{\beta_v^\flat}$ is the number of $\beta_v^\flat$-steps in $d$: this notion is well defined by Prop. 6.1).

**Theorem 16 (Quantitative Simulation of $\lambda_{\mathtt{sh}}$ in $\lambda_{\mathtt{vsub}}$).** *Let $t, u \in \Lambda$. If $d\colon t \to_{\mathtt{sh}}^* u$ then there are $s \in \Lambda_{\mathtt{vsub}}$ and $e\colon t \to_{\mathtt{vsub}}^* s$ such that* Proof p. 41

1. Qualitative Relationship*: $s \equiv \mathtt{m}(u)$;*
2. Quantitative Relationship (Exponential Steps)*: $|d|_{\beta_v^\flat} = |e|_{\mathtt{e}}$;*
3. Normal Form*: if $u$ is* sh-*normal then $s$ and $\mathtt{m}(u)$ are* vsub-*normal.*

**Corollary 17 (Termination Equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{sh}}$).** *Let $t \in \Lambda$. There is a* sh-*normalizing derivation $d$ from $t$ iff there is a* vsub-*normalizing derivation $e$ from $t$. Moreover, $|d|_{\beta_v^\flat} = |e|_{\mathtt{e}}$.* Proof p. 41

As for Cor. 13, the claim of Cor. 17 is stronger than it seems, since for both $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{sh}}$, given a term $t$, if there is a normalizing derivation from $t$ then there are no diverging derivations from $t$ (for $\lambda_{\mathtt{vsub}}$ it follows from strong confluence, for $\lambda_{\mathtt{sh}}$ is given by Prop. 6.4).

About the quantitative relationship, $|d|_{\beta_v^\flat} = |e|_{\mathtt{e}}$ also holds for *all* normalizing derivations from a given term; for $\lambda_{\mathtt{vsub}}$, it holds by Prop. 4.3; for $\lambda_{\mathtt{sh}}$, it is given by the following corollary of Thm. 16.

**Corollary 18 (Number of $\beta_v^\flat$-Steps is Invariant).** *All* sh-*normalizing derivations from $t \in \Lambda$ (if any) have the same number of $\beta_v^\flat$-steps.* Proof p. 41

In a way, the quantitative simulation of $\lambda_{\mathtt{sh}}$ in $\lambda_{\mathtt{vsub}}$ (Thm. 16) "imposes the good behavior" of $\lambda_{\mathtt{vsub}}$ on $\lambda_{\mathtt{sh}}$. The existence of a quantitative invariant in sh-normalizing derivations is not obvious, indeed, as $\lambda_{\mathtt{sh}}$ is not strongly confluent.

Concerning the cost model, things are subtler for $\lambda_{\mathtt{sh}}$. Note that the relationship between $\lambda_{\mathtt{sh}}$ and $\lambda_{\mathtt{vsub}}$ uses the number of $\mathtt{e}$-steps, while the cost model (inherited from $\lambda_{\mathtt{fire}}$) is the number of $\mathtt{m}$-steps. Do $\mathtt{e}$-steps provide a reasonable cost model? Probably not, because there is a family of terms that evaluate in exponentially more $\mathtt{m}$-steps than $\mathtt{e}$-steps. Details are left to a longer version.

## 4 Quantitative Equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{vseq}}$, via $\lambda_{\mathtt{vsub}_k}$

The quantitative termination equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{vseq}}$ is shown in two steps: first, we identify a sub-calculus $\lambda_{\mathtt{vsub}_k}$ of $\lambda_{\mathtt{vsub}}$ equivalent to the whole of $\lambda_{\mathtt{vsub}}$, and then show that $\lambda_{\mathtt{vsub}_k}$ and $\lambda_{\mathtt{vseq}}$ are equivalent (actually isomorphic).

*Equivalence of* $\lambda_{\mathtt{vsub}_k}$ *and* $\lambda_{\mathtt{vsub}}$. The kernel $\lambda_{\mathtt{vsub}_k}$ of $\lambda_{\mathtt{vsub}}$ is the sublanguage of $\lambda_{\mathtt{vsub}}$ obtained by replacing the application constructor $tu$ with the restricted form $tv$ where the right subterm can only be a value $v$—*i.e.*, $\lambda_{\mathtt{vsub}_k}$ is the language of so-called *administrative normal form* [30] of $\lambda_{\mathtt{vsub}}$. The rewriting rules are the same of $\mathtt{vsub}$. It is easy to see that $\mathtt{vsub}_k$ is stable by reduction. For lack of space, more details about $\mathtt{vsub}_k$ have been moved to Appendix B.3 (page 42).

The translation $(\cdot)^+$ of $\lambda_{\mathtt{vsub}}$ into $\lambda_{\mathtt{vsub}_k}$, which simply places the argument of an application into an ES, is defined by (note that $\mathtt{fv}(t) = \mathtt{fv}(t^+)$ for all $t \in \Lambda_{\mathtt{vsub}}$):

$$x^+ := x \qquad (tu)^+ := (t^+x)[x \leftarrow u^+] \ \text{ where } x \notin \mathtt{fv}(t) \cup \mathtt{fv}(u)$$
$$(\lambda x.t)^+ := \lambda x.t^+ \qquad t[x \leftarrow u]^+ := t^+[x \leftarrow u^+]$$

Proof p. 42
**Lemma 19 (Simulation).** *Let $t, u \in \Lambda_{\mathtt{vsub}}$.*
1. Multiplicative*: if $t \to_{\mathtt{m}} u$ then $t^+ \to_{\mathtt{m}} \to_{\mathtt{e}_y} \equiv u^+$;*
2. Exponential*: if $t \to_{\mathtt{e}_\lambda} u$ then $t^+ \to_{\mathtt{e}_\lambda} u^+$, and if $t \to_{\mathtt{e}_y} u$ then $t^+ \to_{\mathtt{e}_y} u^+$.*
3. Structural Equivalence*: $t \equiv u$ implies $t^+ \equiv u^+$.*

Proof p. 44
**Theorem 20 (Quantitative Simulation of $\lambda_{\mathtt{vsub}}$ in $\lambda_{\mathtt{vsub}_k}$).** *Let $t, u \in \Lambda_{\mathtt{vsub}}$. If $d \colon t \to^*_{\mathtt{vsub}} u$ then there are $s \in \Lambda_{\mathtt{vsub}_k}$ and $e \colon t^+ \to^*_{\mathtt{vsub}_k} s$ such that*
1. Qualitative Relationship*: $s \equiv u^+$;*
2. Quantitative Relationship*:*
   1. Multiplicative Steps*: $|e|_{\mathtt{m}} = |d|_{\mathtt{m}}$;*
   2. Exponential Steps*: $|e|_{\mathtt{e}_\lambda} = |d|_{\mathtt{e}_\lambda}$ and $|e|_{\mathtt{e}_y} = |d|_{\mathtt{e}_y} + |d|_{\mathtt{m}}$;*
3. Normal Form*: if $u$ is normal then $s$ is $\mathtt{m}$-normal and $\mathtt{e}(s)$ is normal.*

**Corollary 21 (Linear Termination Equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{vsub}_k}$).** *Let*
Proof p. 44
*$t \in \Lambda_{\mathtt{vsub}}$. There exists a $\mathtt{vsub}$-normalizing derivation $d$ from $t$ iff there exists a $\mathtt{vsub}_k$-normalizing derivation $e$ from $t^+$. Moreover, $|d|_{\mathtt{vsub}} \leq |e|_{\mathtt{vsub}_k} \leq 3|d|_{\mathtt{vsub}}$.*

*Equivalence of* $\lambda_{\mathtt{vsub}_k}$ *and* $\lambda_{\mathtt{vseq}}$. The translation $\underline{\cdot}$ of $\lambda_{\mathtt{vsub}_k}$ into $\lambda_{\mathtt{vseq}}$ is defined as follows, and relies on an auxiliary translation $(\cdot)^\bullet$ of values:

$$x^\bullet := x \qquad (\lambda x.t)^\bullet := \lambda x.\underline{t}$$
$$\underline{v} := \langle v \,|\, \epsilon \rangle \qquad \underline{tv} := \underline{t}@(v^\bullet \cdot \epsilon) \qquad \underline{t[x \leftarrow u]} := \underline{u}@\tilde{\mu}x.\underline{t}$$

It is not hard to see that $\lambda_{\mathtt{vsub}_k}$ and $\lambda_{\mathtt{vseq}}$ are actually isomorphic, where the converse translation $(\cdot)^\diamond$, that maps values and commands to terms, and environments to evaluation contexts, is given by:

$$\begin{array}{rclcrclcrcl} x^\diamond & := & x & \quad & \epsilon^\diamond & := & \langle\cdot\rangle & \quad & \langle v \,|\, e\rangle^\diamond & := & e^\diamond\langle v^\diamond\rangle \\ (\lambda x.c)^\diamond & := & \lambda x.c^\diamond & & (v\cdot e)^\diamond & := & e^\diamond\langle\langle\cdot\rangle v^\diamond\rangle & & (\tilde{\mu}x.c)^\diamond & := & c^\diamond[x{\leftarrow}\langle\cdot\rangle] \end{array}$$

We follow, however, the same structure of the other weaker equivalences.

**Lemma 22 (Simulation of $\to_{\mathtt{vsub}_k}$ by $\to_{\mathtt{vseq}}$).** *Let $t$ and $u$ be $\mathtt{vsub}_k$-terms.* Proof p. 46

1. Multiplicative*: if $t \to_{\mathsf{m}} u$ then $\underline{t} \to_{\bar{\lambda}} \underline{u}$.*
2. Exponential*: if $t \to_{\mathsf{e}} u$ then $\underline{t} \to_{\tilde{\mu}} \underline{u}$.*

**Theorem 23 (Quantitative Simulation of $\lambda_{\mathtt{vsub}_k}$ in $\lambda_{\mathtt{vseq}}$).** *Let $t$ and $u$ be $\mathtt{vsub}_k$-terms. If $d\colon t \to^*_{\mathtt{vsub}_k} u$ then there is $e\colon \underline{t} \to^*_{\mathtt{vseq}} \underline{u}$ such that* Proof p. 47

1. Multiplicative Steps*: $|d|_{\mathsf{m}} = |e|_{\bar{\lambda}}$ (the number $\bar{\lambda}$-steps in $e$);*
2. Exponential Steps*: $|d|_{\mathsf{e}} = |e|_{\tilde{\mu}}$ (the number $\tilde{\mu}$-steps in $e$), so $|d|_{\mathtt{vsub}_k} = |e|_{\mathtt{vseq}}$;*
3. Normal Form*: if $u$ is $\mathtt{vsub}_k$-normal then $\underline{u}$ is $\mathtt{vseq}$-normal.*

**Corollary 24 (Linear Termination Equivalence of $\lambda_{\mathtt{vsub}_k}$ and $\lambda_{\mathtt{vseq}}$).** *Let $t$ be a $\mathtt{vsub}_k$-term. There is a $\mathtt{vsub}_k$-normalizing derivation $d$ from $t$ iff there is a $\mathtt{vseq}$-normalizing derivation $e$ from $\underline{t}$. Moreover, $|d|_{\mathtt{vsub}_k} = |e|_{\mathtt{vseq}}$, $|d|_{\mathsf{e}} = |e|_{\tilde{\mu}}$ and $|d|_{\mathsf{m}} = |e|_{\bar{\lambda}}$.* Proof p. 47

*Structural Equivalence for $\lambda_{\mathtt{vseq}}$.* The equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{vsub}_k}$ relies on the structural equivalence $\equiv$ of $\lambda_{\mathtt{vsub}}$, so it is natural to wonder how does $\equiv$ look on $\lambda_{\mathtt{vseq}}$. The structural equivalence $\doteqdot$ of $\lambda_{\mathtt{vseq}}$ is defined as the closure by evaluation contexts of the following axiom

$$D\langle\tilde{\mu}x.D'\langle\tilde{\mu}y.c\rangle\rangle \doteqdot_{\tilde{\mu}\tilde{\mu}} D'\langle\tilde{\mu}y.D\langle\tilde{\mu}x.c\rangle\rangle \qquad \text{where } x \notin \mathtt{fv}(D') \text{ and } y \notin \mathtt{fv}(D).$$

As expected, $\doteqdot$ has, with respect to $\lambda_{\mathtt{vseq}}$, all the properties of $\equiv$ (see Lemma 5). They are formally stated in the appendix, for lack of space.

## References


1. Accattoli, B., Sacerdoti Coen, C.: On the Relative Usefulness of Fireballs. In: LICS 2015. pp. 141–155 (2015)
2. Accattoli, B.: Proof nets and the call-by-value λ-calculus. Theor. Comput. Sci. 606, 2–24 (2015)
3. Accattoli, B., Paolini, L.: Call-by-Value Solvability, revisited. In: FLOPS. pp. 4–16 (2012)
4. Accattoli, B., Sacerdoti Coen, C.: On the Value of Variables. In: WoLLIC 2014. pp. 36–50 (2014)
5. Ariola, Z.M., Bohannon, A., Sabry, A.: Sequent calculi and abstract machines. ACM Trans. Program. Lang. Syst. 31(4) (2009)
6. Barendregt, H.P.: The Lambda Calculus – Its Syntax and Semantics, vol. 103. North-Holland (1984)

7. Blelloch, G.E., Greiner, J.: Parallelism in Sequential Functional Languages. In: FPCA. pp. 226–237 (1995)
8. Carraro, A., Guerrieri, G.: A Semantical and Operational Account of Call-by-Value Solvability. In: FOSSACS 2014. pp. 103–118 (2014)
9. Curien, P.L., Herbelin, H.: The duality of computation. In: ICFP. pp. 233–243 (2000)
10. Curien, P., Munch-Maccagnoni, G.: The Duality of Computation under Focus. In: 6th IFIP, TCS 2010. Proceedings. vol. 323, pp. 165–181. Springer (2010)
11. Dal Lago, U., Martini, S.: The weak lambda calculus as a reasonable machine. Theor. Comput. Sci. 398(1-3), 32–50 (2008)
12. Dyckhoff, R., Lengrand, S.: Call-by-Value lambda-calculus and LJQ. J. Log. Comput. 17(6), 1109–1134 (2007)
13. Grégoire, B., Leroy, X.: A compiled implementation of strong reduction. In: (ICFP '02). pp. 235–246 (2002)
14. Guerrieri, G.: Head reduction and normalization in a call-by-value lambda-calculus. In: WPTE 2015. pp. 3–17 (2015)
15. Guerrieri, G., Paolini, L., Ronchi Della Rocca, S.: Standardization of a Call-By-Value Lambda-Calculus. In: TLCA 2015. pp. 211–225 (2015)
16. Herbelin, H.: C'est maintenant qu'on calcule: au cœur de la dualité (2005), Habilitation thesis
17. Herbelin, H., Zimmermann, S.: An operational account of Call-by-Value Minimal and Classical $\lambda$-calculus in Natural Deduction form. In: TLCA. pp. 142–156 (2009)
18. Jones, N.D., Gomard, C.K., Sestoft, P.: Partial Evaluation and Automatic Program Generation. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1993)
19. Lassen, S.: Eager Normal Form Bisimulation. In: LICS 2005. pp. 345–354 (2005)
20. Lévy, J.J.: Réductions correctes et optimales dans le lambda-calcul. Thése d'Etat, Univ. Paris VII, France (1978)
21. Maraist, J., Odersky, M., Turner, D.N., Wadler, P.: Call-by-name, Call-by-value, Call-by-need and the Linear $\lambda$-Calculus. TCS 228(1-2), 175–210 (1999)
22. Moggi, E.: Computational $\lambda$-Calculus and Monads. In: LICS '89. pp. 14–23 (1989)
23. Paolini, L., Pimentel, E., Ronchi Della Rocca, S.: Strong Normalization from an unusual point of view. Theor. Comp. Science 412(20), 1903–1915 (2011)
24. Paolini, L.: Call-by-Value Separability and Computability. In: ICTCS. pp. 74–89 (2002)
25. Paolini, L., Pimentel, E., Ronchi Della Rocca, S.: Lazy strong normalization. In: ITRS '04. Electronic Notes in Theoretical Computer Science, vol. 136C, pp. 103–116 (2005)
26. Paolini, L., Ronchi Della Rocca, S.: Call-by-value Solvability. ITA 33(6), 507–534 (1999)
27. Plotkin, G.D.: Call-by-Name, Call-by-Value and the lambda-Calculus. Theor. Comput. Sci. 1(2), 125–159 (1975)
28. Regnier, L.: Une équivalence sur les lambda-termes. TCS 2(126), 281–292 (1994)
29. Ronchi Della Rocca, S., Paolini, L.: The Parametric $\lambda$-Calculus. Springer Berlin Heidelberg (2004)
30. Sabry, A., Felleisen, M.: Reasoning about Programs in Continuation-Passing Style. Lisp and Symbolic Computation 6(3-4), 289–360 (1993)
31. Sabry, A., Wadler, P.: A Reflection on Call-by-Value. ACM Trans. Program. Lang. Syst. 19(6), 916–941 (1997)
32. Sands, D., Gustavsson, J., Moran, A.: Lambda Calculi and Linear Speedups. In: The Essence of Computation, Complexity, Analysis, Transformation. Essays Dedicated to Neil D. Jones. pp. 60–84 (2002)

# Technical Appendix

## A Rewriting Theory: Definitions, Notations, and Basic Results

Given a binary relation $\to_{\mathsf{r}}$ on a set $I$, the reflexive-transitive (resp. reflexive; transitive; reflexive-transitive and symmetric) closure of $\to_{\mathsf{r}}$ is denoted by $\to^*$ (resp. $\to_{\mathsf{r}}^=$; $\to_{\mathsf{r}}^+$; $\simeq_{\mathsf{r}}$). The transpose of $\to_{\mathsf{r}}$ is denoted by ${}_{\mathsf{r}}\!\leftarrow$. A ($\mathsf{r}$-)*derivation* $d$ *from* $t$ *to* $u$, denoted by $d\colon t \to_{\mathsf{r}}^* u$, is a finite sequence $(t_i)_{0 \le i \le n}$ of elements of $I$ (with $n \in \mathbb{N}$) s.t. $t = t_0$, $u = t_n$ and $t_i \to_{\mathsf{r}} t_{i+1}$ for all $1 \le i < n$;

The *number of* $\mathsf{r}$*-steps of a derivation* $d$, *i.e.* its *length*, is denoted by $|d|_{\mathsf{r}} \coloneqq n$, or simply $|d|$. If $\to_{\mathsf{r}} = \to_1 \cup \to_2$ with $\to_1 \cap \to_2 = \emptyset$, $|d|_i$ is the number of $\to_i$-steps in $d$, for $i = 1, 2$. We say that:

- $t \in I$ is $\mathsf{r}$-*normal* or a $\mathsf{r}$-*normal form* if $t \not\to_{\mathsf{r}} u$ for all $u \in I$; $u \in I$ is a $\mathsf{r}$-*normal form of* $t$ if $u$ is $\mathsf{r}$-normal and $t \to_{\mathsf{r}}^* u$;
- $t \in I$ is $\mathsf{r}$-*normalizable* if there is a $\mathsf{r}$-normal $u \in I$ s.t. $t \to_{\mathsf{r}}^* u$; $t$ is *strongly* $\mathsf{r}$-*normalizable* if there is no infinite sequence $(t_i)_{i \in \mathbb{N}}$ s.t. $t_0 = t$ and $t_i \to_{\mathsf{r}} t_{i+1}$;
- a $\mathsf{r}$-derivation $d\colon t \to_{\mathsf{r}}^* u$ is ($\mathsf{r}$-)*normalizing* if $u$ is $\mathsf{r}$-normal;
- $\to_{\mathsf{r}}$ is *strongly normalizing* if all $t \in I$ is strongly $\mathsf{r}$-normalizable;
- $\to_{\mathsf{r}}$ is *strongly confluent* if, for all $t, u, s \in I$ s.t. $s \;{}_{\mathsf{r}}\!\leftarrow t \to_{\mathsf{r}} u$ and $u \neq s$, there is $r \in I$ s.t. $s \to_{\mathsf{r}} r \;{}_{\mathsf{r}}\!\leftarrow u$; $\to_{\mathsf{r}}$ is *confluent* if $\to_{\mathsf{r}}^*$ is strongly confluent.

Let $\to_1, \to_2 \subseteq I \times I$. Composition of relations is denoted by juxtaposition: for instance, $t \to_1\to_2 u$ means that there is $s \in I$ s.t. $t \to_1 s \to_2 u$; for any $n \in \mathbb{N}$, $t \to_1^n u$ means that there is a $\to_1$-derivation with length $n$ ($t = u$ for $n = 0$). We say that $\to_1$ and $\to_2$ *strongly commute* if, for any $t, u, s \in I$ s.t. $u \;{}_1\!\leftarrow t \to_2 s$, one has $u \neq s$ and there is $r \in I$ s.t. $u \to_2 r \;{}_1\!\leftarrow s$. Note that if $\to_1$ and $\to_2$ strongly commute and $\to = \to_1 \cup \to_2$, then for any derivation $d\colon t \to^* u$ the sizes $|d|_1$ and $|d|_2$ are uniquely determined.

The following proposition collects some basic and well-known results of rewriting theory.

**Proposition 25.** *Let $\to_{\mathsf{r}}$ be a binary relation on a set $I$.*

1. *If $\to_{\mathsf{r}}$ is confluent then:*
   (a) *every $\mathsf{r}$-normalizable term has a unique $\mathsf{r}$-normal form;*
   (b) *for all $t, u \in I$, $t \simeq_{\mathsf{r}} u$ iff there is $s \in I$ s.t. $t \to_{\mathsf{r}}^* s \;{}_{\mathsf{r}}^*\!\leftarrow u$.*
2. *If $\to_{\mathsf{r}}$ is strongly confluent then $\to_{\mathsf{r}}$ is confluent and, for any $t \in I$, one has:*
   (a) *all normalizing $\mathsf{r}$-derivations from $t$ have the same length;*
   (b) *$t$ is strongly $\mathsf{r}$-normalizable if and only if $t$ is $\mathsf{r}$-normalizable.*

As all incarnations of Open CBV we consider are confluent, the use of Prop. 25.1 is left implicit.

For $\lambda_{\mathtt{fire}}$ and $\lambda_{\mathtt{vsub}}$, we use Prop. 25.2 and the following more informative version of Hindley–Rosen Lemma, whose proof is just a more accurate reading of the proof in [6, Prop. 3.3.5.(i)]:

**Lemma 26 (Strong Hindley–Rosen).** *Let $\to = \to_1 \cup \to_2$ be a binary relation on a set $I$ s.t. $\to_1$ and $\to_2$ are strongly confluent. If $\to_1$ and $\to_2$ strongly commute, then $\to$ is strongly confluent and, for any $t \in I$ and any normalizing derivations $d$ and $e$ from $t$, one has $|d| = |e|$, $|d|_1 = |e|_1$ and $|d|_2 = |e|_2$.*

# B Omitted Proofs

## B.1 Proofs of Section 2 (Incarnations of Open Call-by-Value)

*Naïve Open CBV: Plotkin's Calculus $\lambda_{\texttt{Plot}}$*

*Remark 27.* Since $\to_{\beta_v}$ does not reduce under $\lambda$'s, any value is $\beta_v$-normal, and so $\beta_y$-normal and $\beta_\lambda$-normal, as $\to_{\beta_y}, \to_{\beta_\lambda} \subseteq \to_{\beta_v}$.

See p. 5

**Proposition 1.** *$\to_{\beta_y}$, $\to_{\beta_\lambda}$ and $\to_{\beta_v}$ are strongly confluent.*

*Proof.* We prove that $\to_{\beta_v}$ is strongly confluent. The proofs that $\to_{\beta_y}$ and $\to_{\beta_\lambda}$ are strongly confluent are perfectly analogous.

So, we prove, by induction on $t$, that if $t \to_{\beta_v} u$ and $t \to_{\beta_v} s$ with $u \neq s$, then there exists $t'$ such that $u \to_{\beta_v} t'$ and $s \to_{\beta_v} t'$.

Observe that neither $t \to_{\beta_v} u$ nor $t \to_{\beta_v} s$ can be a step at the root: indeed, if $t \coloneqq (\lambda x.r)v \to_{\beta_v} r\{x{\leftarrow}v\} =: u$ and $t \to_{\beta_v} s$ (or if $t \coloneqq (\lambda x.r)v \to_{\beta_v} r\{x{\leftarrow}v\} =: s$ and $t \to_{\beta_v} u$), then $u = s$ since $\lambda x.r$ and $v$ are $\beta_v$-normal by Remark 27; but this contradicts the hypothesis $u \neq s$. So, according to the definition of $t \to_{\beta_v} u$ and $t \to_{\beta_v} s$, there are only four cases.

- *Application Left for $t \to_{\beta_v} u$ and $t \to_{\beta_v} s$, i.e.* $t = rq \to_{\beta_v} pq = u$ and $t = rq \to_{\beta_v} mq = s$ with $r \to_{\beta_v} p$ and $r \to_{\beta_v} m$. By the hypothesis $u \neq s$ it follows that $p \neq m$. By *i.h.*, there exists $r'$ such that $p \to_{\beta_v} r'$ and $m \to_{\beta_v} r'$. So, setting $t' = r'q$, one has $u = pq \to_{\beta_v} t'$ and $s = mq \to_{\beta_v} t'$.
- *Application Right for $t \to_{\beta_v} u$ and $t \to_{\beta_v} s$, i.e.* $t = rq \to_{\beta_v} rp = u$ and $t = rq \to_{\beta_v} rm = s$ with $q \to_{\beta_v} p$ and $q \to_{\beta_v} m$. From the hypothesis $u \neq s$ it follows that $p \neq m$. By *i.h.*, there exists $q'$ such that $p \to_{\beta_v} q'$ and $m \to_{\beta_v} q'$. So, setting $t' = rq'$, one has $u = rp \to_{\beta_v} t'$ and $s = rm \to_{\beta_v} t'$.
- *Application Left for $t \to_{\beta_v} u$* and *Application Right for $t \to_{\beta_v} s$, i.e.* $t = rq \to_{\beta_v} pq = u$ and $t = rq \to_{\beta_v} rm = s$ with $r \to_{\beta_v} p$ and $q \to_{\beta_v} m$. So, setting $t' = pm$, one has $u = pq \to_{\beta_v} t'$ and $s = rm \to_{\beta_v} t'$.
- *Application Right for $t \to_{\beta_v} u$* and *Application Left for $t \to_{\beta_v} s$, i.e.* $t = rq \to_{\beta_v} rp = u$ and $t = rq \to_{\beta_v} mq = s$ with $q \to_{\beta_v} p$ and $r \to_{\beta_v} m$. So, setting $t' = mp$, one has $u = rp \to_{\beta_v} t'$ and $s = mq \to_{\beta_v} t'$. □

*Open CBV 1: the Fireball Calculus $\lambda_{\texttt{fire}}$*

**Lemma 28 (Values and inert terms are $\beta_f$-normal).**

1. *Every value is $\beta_f$-normal.*
2. *Every inert term is $\beta_f$-normal.*

*Proof.*

1. Immediate, since $\to_{\beta_f}$ does not reduce under $\lambda$'s.
2. By induction on the definition of inert term $i$.
   - If $i = x$ then $i$ is obviously $\beta_f$-normal.
   - If $i = i'\lambda x.t$ then $i'$ and $\lambda x.t$ are $\beta_f$-normal by *i.h.* and Lemma 28.1 respectively, besides $i'$ is not an abstraction, so $i$ is $\beta_f$-normal.
   - Finally, if $i = i'i''$ then $i'$ and $i''$ are $\beta_f$-normal by *i.h.*, moreover $i'$ is not an abstraction, hence $i$ is $\beta_f$-normal. □

**Proposition 2** (Open Harmony). *Let $t \in \Lambda$: $t$ is $\beta_f$-normal iff $t$ is a fireball.* See p. 6

*Proof.*

**⇒:** Proof by induction on $t \in \Lambda$. If $t$ is a value then $t$ is a fireball.
Otherwise $t = us$ for some terms $u$ and $s$. Since $t$ is $\beta_f$-normal, then $u$ and $s$ are $\beta_f$-normal, and either $u$ is not an abstraction or $s$ is not a fireball. By induction hypothesis, $u$ and $s$ are fireballs. Summing up, $u$ is either a variable or an inert term, and $s$ is a fireball, therefore $t = us$ is an inert term and hence a fireball.

**⇐:** By hypothesis, $t$ is either a value or an inert term. If $t$ is a value, then it is $\beta_f$-normal by Lemma 28.1. Otherwise $t$ is an inert term and then it is $\beta_f$-normal by Lemma 28.2. □

**Lemma 29.** *For every $t, t' \in \Lambda$, if $t \to_{\beta_i} t'$ then $t \neq t'$.*

*Proof.* By induction on $t \in \Lambda$. According to the definition of $t \to_{\beta_i} t'$, there are three cases.

- *Step at the root*, *i.e.* $t = (\lambda x.u)i \to_{\beta_i} u\{x{\leftarrow}i\} = t'$: then, since $i$ is not an abstraction, necessarily $t = (\lambda x.u)i \neq u\{x{\leftarrow}i\} = t'$.
- *Application Left*, *i.e.* $t = us \to_{\beta_i} u's = t'$ with $u \to_{\beta_i} u'$: by *i.h.*, $u \neq u'$ and hence $t = us \neq u's = t'$.
- *Application Right*, *i.e.* $t = us \to_{\beta_i} us' = t'$ with $s \to_{\beta_i} s'$: by *i.h.*, $s \neq s'$ and hence $t = us \neq us' = t'$. □

**Proposition 3** (Basic Properties of $\lambda_{\mathtt{fire}}$). See p. 7

1. *$\to_{\beta_i}$ is strongly normalizing and strongly confluent.*
2. *$\to_{\beta_\lambda}$ and $\to_{\beta_i}$ strongly commute.*
3. *$\to_{\beta_f}$ is strongly confluent, and all $\beta_f$-normalizing derivations $d$ from $t \in \Lambda$ (if any) have the same length $|d|_{\beta_f}$, the same number $|d|_{\beta_\lambda}$ of $\beta_\lambda$-steps, and the same number $|d|_{\beta_i}$ of $\beta_i$-steps.*

*Proof.*

1. Strong normalization of $\to_{\beta_i}$ follows from general termination properties in the ordinary (*i.e.* pure, strong, and call-by-name) $\lambda$-calculus, as we now explain. Since $\beta_i$-steps do not substitute abstractions, they can only cause creations of type 1, according to Lévy's classification of creations of $\beta$-redexes [20]. Then $\beta_i$-derivations can be seen as special cases of $\mathsf{m}$-*developments* (see

Accattoli, B., Kesner, D., *The Permutative $\lambda$-Calculus.* In: LPAR. pp. 23-36, 2012), in turn a special case of more famous *superdevelopments*, *i.e.* reduction sequences reducing only (residuals of) redexes in the original term plus creations of type 1 (m-developments) or type 1 and 2 (superdevelopments). Both m-developments and superdevelopments always terminate. Therefore, $\rightarrow_{\beta_i}$ is strongly normalizing.

Now, we prove that $\rightarrow_{\beta_i}$ is strongly confluent, that is if $t \rightarrow_{\beta_i} u$ and $t \rightarrow_{\beta_i} s$ with $u \neq s$, then there exists $t' \in \Lambda$ such that $u \rightarrow_{\beta_i} t'$ and $s \rightarrow_{\beta_i} t'$. The proof is by induction on $t \in \Lambda$.
Observe that neither $t \rightarrow_{\beta_i} u$ nor $t \rightarrow_{\beta_i} s$ can be a step at the root: indeed, if $t \coloneqq (\lambda x.r)i \mapsto_{\beta_i} r\{x{\leftarrow}i\} \coloneqq u$ and $t \rightarrow_{\beta_i} s$ (or if $t \coloneqq (\lambda x.r)i \mapsto_{\beta_i} r\{x{\leftarrow}i\} =: s$ and $t \rightarrow_{\beta_i} u$), then $u = s$ since $\lambda x.r$ and $i$ are $\beta_i$-normal by Lemmas 28.1-2 (as $\rightarrow_{\beta_i} \subseteq \rightarrow_{\beta_f}$); but this contradicts the hypothesis $u \neq s$. So, according to the definition of $t \rightarrow_{\beta_i} u$ and $t \rightarrow_{\beta_i} s$, there are only four cases.

- *Application Left for $t \rightarrow_{\beta_i} u$ and $t \rightarrow_{\beta_i} s$, i.e.* $t = rq \rightarrow_{\beta_i} pq = u$ and $t = rq \rightarrow_{\beta_i} mq = s$ with $r \rightarrow_{\beta_i} p$ and $r \rightarrow_{\beta_i} m$. By the hypothesis $u \neq s$ it follows that $p \neq m$. By *i.h.*, there exists $r'$ such that $p \rightarrow_{\beta_i} r'$ and $m \rightarrow_{\beta_i} r'$. So, setting $t' = r'q$, one has $u = pq \rightarrow_{\beta_i} t'$ and $s = mq \rightarrow_{\beta_i} t'$.
- *Application Right for $t \rightarrow_{\beta_i} u$ and $t \rightarrow_{\beta_i} s$, i.e.* $t = rq \rightarrow_{\beta_i} rp = u$ and $t = rq \rightarrow_{\beta_i} rm = s$ with $q \rightarrow_{\beta_i} p$ and $q \rightarrow_{\beta_i} m$. By the hypothesis $u \neq s$ it follows that $p \neq m$. By *i.h.*, there exists $q'$ such that $p \rightarrow_{\beta_i} q'$ and $m \rightarrow_{\beta_i} q'$. So, setting $t' = rq'$, one has $u = rp \rightarrow_{\beta_i} t'$ and $s = rm \rightarrow_{\beta_i} t'$.
- *Application Left for $t \rightarrow_{\beta_i} u$* and *Application Right for $t \rightarrow_{\beta_i} s$, i.e.* $t = rq \rightarrow_{\beta_i} pq = u$ and $t = rq \rightarrow_{\beta_i} rm = s$ with $r \rightarrow_{\beta_i} p$ and $q \rightarrow_{\beta_i} m$. So, setting $t' = pm$, one has $u = pq \rightarrow_{\beta_i} t'$ and $s = rm \rightarrow_{\beta_i} t'$.
- *Application Right for $t \rightarrow_{\beta_i} u$* and *Application Left for $t \rightarrow_{\beta_i} s$, i.e.* $t = rq \rightarrow_{\beta_i} rp = u$ and $t = rq \rightarrow_{\beta_i} mq = s$ with $q \rightarrow_{\beta_i} p$ and $r \rightarrow_{\beta_i} m$. So, setting $t' = mp$, one has $u = rp \rightarrow_{\beta_i} t'$ and $s = mq \rightarrow_{\beta_i} t'$.

2. We prove, by induction on $t \in \Lambda$, that if $t \rightarrow_{\beta_\lambda} u$ and $t \rightarrow_{\beta_i} s$, then $u \neq s$ and there is $t' \in \Lambda$ such that $u \rightarrow_{\beta_i} t'$ and $s \rightarrow_{\beta_\lambda} t'$.
Observe that neither $t \rightarrow_{\beta_\lambda} u$ nor $t \rightarrow_{\beta_i} s$ can be a step at the root: indeed, if $t \coloneqq (\lambda x.r)\lambda y.q \mapsto_{\beta_\lambda} r\{x{\leftarrow}\lambda y.q\} =: u$ (resp. $t \coloneqq (\lambda x.r)i \mapsto_{\beta_i} r\{x{\leftarrow}i\} =: s$) then $\lambda y.q$ is not a inert term (resp. $i$ is not an abstraction), moreover $\lambda x.r$ and $\lambda y.q$ (resp. $i$) are $\beta_i$-normal (resp. $\beta_\lambda$-normal) by Prop. 2, as $\rightarrow_{\beta_i} \subseteq \rightarrow_{\beta_f}$ (resp. $\rightarrow_{\beta_\lambda} \subseteq \rightarrow_{\beta_f}$); therefore, $t$ is $\beta_i$-normal (resp. $\beta_\lambda$-normal) but this contradicts the hypothesis $t \rightarrow_{\beta_i} s$ (resp. $t \rightarrow_{\beta_\lambda} u$). So, according to the definitions of $t \rightarrow_{\beta_\lambda} u$ and $t \rightarrow_{\beta_i} s$, there are only four cases.
   - *Application Left for both $t \rightarrow_{\beta_\lambda} u$ and $t \rightarrow_{\beta_i} s$, i.e.* $t \coloneqq rq \rightarrow_{\beta_\lambda} pq =: u$ and $t \coloneqq rq \rightarrow_{\beta_i} mq =: s$ with $r \rightarrow_{\beta_\lambda} p$ and $r \rightarrow_{\beta_i} m$. By *i.h.*, $p \neq m$ and there exists $r'$ such that $p \rightarrow_{\beta_i} r'$ and $m \rightarrow_{\beta_\lambda} r'$. So, $u \neq s$ and, setting $t' \coloneqq r'q$, one has $u = pq \rightarrow_{\beta_i} t' \;{}_{\beta_\lambda}\!\!\leftarrow mq = s$.
   - *Application Right for both $t \rightarrow_{\beta_\lambda} u$ and $t \rightarrow_{\beta_i} s$, i.e.* $t \coloneqq rq \rightarrow_{\beta_\lambda} rp =: u$ and $t \coloneqq rq \rightarrow_{\beta_i} rm =: s$ with $q \rightarrow_{\beta_\lambda} p$ and $q \rightarrow_{\beta_i} m$. By *i.h.*, $p \neq m$ and there exists $q'$ such that $p \rightarrow_{\beta_i} q'$ and $m \rightarrow_{\beta_\lambda} q'$. So, $u \neq s$ and, setting $t' \coloneqq rq'$, one has $u = rp \rightarrow_{\beta_i} t' \;{}_{\beta_\lambda}\!\!\leftarrow rm = s$.

– *Application Left for* $t \to_{\beta_\lambda} u$ and *Application Right for* $t \to_{\beta_i} s$, *i.e.* $t := rq \to_{\beta_\lambda} pq = u$ and $t = rq \to_{\beta_i} rm =: s$ with $r \to_{\beta_\lambda} p$ and $q \to_{\beta_i} m$. By Lemma 29, $q \neq m$ and hence $u = pq \neq rm = s$. Setting $t' := pm$, one has $u = pq \to_{\beta_i} t' \; {}_{\beta_\lambda}\!\leftarrow rm = s$.

– *Application Right for* $t \to_{\beta_\lambda} u$ and *Application Left for* $t \to_{\beta_i} s$, *i.e.* $t := rq \to_{\beta_\lambda} rp =: u$ and $t = rq \to_{\beta_i} mq = s$ with $q \to_{\beta_\lambda} p$ and $r \to_{\beta_i} m$. By Lemma 29, $r \neq m$ and hence $u = rp \neq mq = s$. Setting $t' := mp$, one has $u = rp \to_{\beta_i} t' \; {}_{\beta_\lambda}\!\leftarrow mq = s$.

3. It follows immediately from strong confluence of $\to_{\beta_\lambda}$ (Prop. 1.1) and $\to_{\beta_i}$ (Prop. 3.1), the strong commutation of $\to_{\beta_\lambda}$ and $\to_{\beta_i}$ (Prop. 3.2), and Hindley-Rosen (Lemma 26). □

*Open CBV 2: the Value Substitution Calculus* $\lambda_{\mathtt{vsub}}$

**Proposition 4** (Basic Properties of $\lambda_{\mathtt{vsub}}$, [3]). See p. 8

*1.* $\to_{\mathtt{m}}$ *and* $\to_{\mathtt{e}}$ *are strongly normalizing (separately).*
*2.* $\to_{\mathtt{m}}$ *and* $\to_{\mathtt{e}}$ *are strongly confluent (separately).*
*3.* $\to_{\mathtt{m}}$ *and* $\to_{\mathtt{e}}$ *strongly commute.*
*4.* $\to_{\mathtt{vsub}}$ *is strongly confluent, and all* vsub*-normalizing derivations* $d$ *from* $t \in \Lambda_{\mathtt{vsub}}$ *(if any) have the same length* $|d|_{\mathtt{vsub}}$*, the same number* $|d|_{\mathtt{e}}$ *of* e*-steps, and the same number* $|d|_{\mathtt{e}}$ *of* m*-steps.*
*5. Let* $t \in \Lambda$*. For any* vsub*-derivation* $d$ *from* $t$*,* $|d|_{\mathtt{e}} \leq |d|_{\mathtt{m}}$*.*

*Proof.* The statements of Prop. 4 are a refinement of some results proved in [3], where $\to_{\mathtt{vsub}}$ is denoted by $\to_{\mathtt{w}}$.

1. In [3, Lemma 3] it has been proved that $\to_{\mathtt{dB}}$ and $\to_{\mathtt{vs}}$ are strongly normalizing, separately. Since $\to_{\mathtt{m}} \subseteq \to_{\mathtt{dB}}$ and $\to_{\mathtt{e}} \subseteq \to_{\mathtt{vs}}$ ($\to_{\mathtt{dB}}$ and $\to_{\mathtt{vs}}$ are just the extensions of $\to_{\mathtt{m}}$ and $\to_{\mathtt{e}}$, respectively, obtained by allowing reductions under $\lambda$'s), one has that $\to_{\mathtt{m}}$ and $\to_{\mathtt{e}}$ are strongly normalizing, separately.
2. We prove that $\to_{\mathtt{m}}$ is strongly confluent, *i.e.* if $u \;{}_{\mathtt{m}}\!\leftarrow t \to_{\mathtt{m}} s$ with $u \neq s$ then there exists $t' \in \Lambda_{\mathtt{vsub}}$ such that $u \to_{\mathtt{m}} t' \;{}_{\mathtt{m}}\!\leftarrow s$. The proof is by induction on the definition of $\to_{\mathtt{m}}$. Since there $t \to_{\mathtt{m}} s \neq u$ and the reduction $\to_{\mathtt{m}}$ is weak, there are only eight cases:
   – *Step at the Root for* $t \to_{\mathtt{m}} u$ *and Application Right for* $t \to_{\mathtt{m}} s$, *i.e.* $t := L\langle \lambda x.q \rangle r \mapsto_{\mathtt{m}} L\langle q[x{\leftarrow}r] \rangle =: u$ and $t \mapsto_{\mathtt{m}} L\langle \lambda x.q \rangle r' =: s$ with $r \to_{\mathtt{m}} r'$: then, $u \to_{\mathtt{m}} L\langle q[x{\leftarrow}r'] \rangle \;{}_{\mathtt{m}}\!\leftarrow s$;
   – *Step at the Root for* $t \to_{\mathtt{m}} u$ *and Application Left for* $t \to_{\mathtt{m}} s$, *i.e.*, for some $n > 0$, $t := (\lambda x.q)[x_1{\leftarrow}t_1] \ldots [x_n{\leftarrow}t_n] r \mapsto_{\mathtt{m}} q[x{\leftarrow}r][x_1{\leftarrow}t_1] \ldots [x_n{\leftarrow}t_n] =: u$ whereas $t \to_{\mathtt{m}} (\lambda x.q)[x_1{\leftarrow}t_1] \ldots [x_j{\leftarrow}t'_j] \ldots [x_n{\leftarrow}t_n] r =: s$ with $t_j \to_{\mathtt{m}} t'_j$ for some $1 \leq j \leq n$: then,
   $$u \to_{\mathtt{m}} q[x{\leftarrow}r][x_1{\leftarrow}t_1] \ldots [x_j{\leftarrow}t'_j] \ldots [x_n{\leftarrow}t_n] \;{}_{\mathtt{m}}\!\leftarrow s;$$
   – *Application Left for* $t \to_{\mathtt{m}} u$ *and Application Right for* $t \to_{\mathtt{m}} s$, *i.e.* $t := rq \to_{\mathtt{m}} r'q =: u$ and $t \to_{\mathtt{m}} rq' =: s$ with $r \to_{\mathtt{m}} r'$ and $q \to_{\mathtt{m}} q'$: then, $u \to_{\mathtt{m}} r'q' \;{}_{\mathtt{m}}\!\leftarrow s$;

- *Application Left for both* $t \to_{\mathsf{m}} u$ *and* $t \to_{\mathsf{m}} s$, *i.e.* $t \coloneqq rq \to_{\mathsf{m}} r'q =: u$ and $t \to_{\mathsf{m}} r''q =: s$ with $r' \ {}_{\mathsf{m}}\!\leftarrow r \to_{\mathsf{m}} r''$: by *i.h.*, there exists $r_0 \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{m}} r_0 \ {}_{\mathsf{m}}\!\leftarrow r''$, hence $u \to_{\mathsf{m}} r_0 q \ {}_{\mathsf{m}}\!\leftarrow s$;
- *Application Right for both* $t \to_{\mathsf{m}} u$ *and* $t \to_{\mathsf{m}} s$, *i.e.* $t \coloneqq qr \to_{\mathsf{m}} qr' =: u$ and $t \to_{\mathsf{m}} qr'' =: s$ with $r' \ {}_{\mathsf{m}}\!\leftarrow r \to_{\mathsf{m}} r''$: by *i.h.*, there exists $r_0 \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{m}} r_0 \ {}_{\mathsf{m}}\!\leftarrow r''$, hence $u \to_{\mathsf{m}} qr_0 \ {}_{\mathsf{m}}\!\leftarrow s$;
- *ES Left for* $t \to_{\mathsf{m}} u$ *and ES Right for* $t \to_{\mathsf{m}} s$, *i.e.* $t \coloneqq r[x{\leftarrow}q] \to_{\mathsf{m}} r'[x{\leftarrow}q] =: u$ and $t \to_{\mathsf{m}} r[x{\leftarrow}q'] =: s$ with $r \to_{\mathsf{m}} r'$ and $q \to_{\mathsf{m}} q'$: then, $u \to_{\mathsf{m}} r'[x{\leftarrow}q'] \ {}_{\mathsf{m}}\!\leftarrow s$;
- *ES Left for both* $t \to_{\mathsf{m}} u$ *and* $t \to_{\mathsf{m}} s$, *i.e.* $t \coloneqq r[x{\leftarrow}q] \to_{\mathsf{m}} r'[x{\leftarrow}q] =: u$ and $t \to_{\mathsf{m}} r''[x{\leftarrow}q] =: s$ with $r' \ {}_{\mathsf{m}}\!\leftarrow r \to_{\mathsf{m}} r''$: by *i.h.*, there exists $r_0 \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{m}} r_0 \ {}_{\mathsf{m}}\!\leftarrow r''$, hence $u \to_{\mathsf{m}} r_0[x{\leftarrow}q] \ {}_{\mathsf{m}}\!\leftarrow s$;
- *ES Right for both* $t \to_{\mathsf{m}} u$ *and* $t \to_{\mathsf{m}} s$, *i.e.* $t \coloneqq q[x{\leftarrow}r] \to_{\mathsf{m}} q[x{\leftarrow}r'] =: u$ and $t \to_{\mathsf{m}} q[x{\leftarrow}r''] =: s$ with $r' \ {}_{\mathsf{m}}\!\leftarrow r \to_{\mathsf{m}} r''$: by *i.h.*, there exists $r_0 \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{m}} r_0 \ {}_{\mathsf{m}}\!\leftarrow r''$, hence $u \to_{\mathsf{m}} q[x{\leftarrow}r_0] \ {}_{\mathsf{m}}\!\leftarrow s$.

We prove that $\to_{\mathsf{e}}$ is strongly confluent, *i.e.* if $u \ {}_{\mathsf{e}}\!\leftarrow t \to_{\mathsf{e}} s$ with $u \neq s$ then there exists $r \in \Lambda_{\mathsf{vsub}}$ such that $u \to_{\mathsf{e}} t' \ {}_{\mathsf{e}}\!\leftarrow s$. The proof is by induction on the definition of $\to_{\mathsf{e}}$. Since there $t \to_{\mathsf{e}} s \neq u$ and the reduction $\to_{\mathsf{e}}$ is weak, there are only eight cases:

- *Step at the Root for* $t \to_{\mathsf{e}} u$ and *ES Left for* $t \to_{\mathsf{e}} s$, *i.e.* $t \coloneqq r[x{\leftarrow}L\langle v\rangle] \mapsto_{\mathsf{e}} L\langle r\{x{\leftarrow}v\}\rangle =: u$ and $t \mapsto_{\mathsf{e}} r'[x{\leftarrow}L\langle v\rangle] =: s$ with $r \to_{\mathsf{e}} r'$: then, $u \to_{\mathsf{e}} L\langle r'[x{\leftarrow}v]\rangle \ {}_{\mathsf{e}}\!\leftarrow s$;
- *Step at the Root for* $t \to_{\mathsf{e}} u$ and *ES Right for* $t \to_{\mathsf{e}} s$, *i.e.*, for some $n > 0$, $t \coloneqq r[x{\leftarrow}v[x_1{\leftarrow}t_1]\dots[x_n{\leftarrow}t_n]] \mapsto_{\mathsf{e}} r\{x{\leftarrow}v\}[x_1{\leftarrow}t_1]\dots[x_n{\leftarrow}t_n] =: u$ whereas $t \to_{\mathsf{e}} r[x{\leftarrow}v[x_1{\leftarrow}t_1]\dots[x_j{\leftarrow}t'_j]\dots[x_n{\leftarrow}t_n]] =: s$ with $t_j \to_{\mathsf{e}} t'_j$ for some $1 \leq j \leq n$: then,
$$u \to_{\mathsf{e}} r\{x{\leftarrow}v\}[x_1{\leftarrow}t_1]\dots[x_j{\leftarrow}t'_j]\dots[x_n{\leftarrow}t_n] \ {}_{\mathsf{e}}\!\leftarrow s;$$
- *Application Left for* $t \to_{\mathsf{e}} u$ and *Application Right for* $t \to_{\mathsf{e}} s$, *i.e.* $t \coloneqq rq \to_{\mathsf{e}} r'q =: u$ and $t \to_{\mathsf{e}} rq' =: s$ with $r \to_{\mathsf{e}} r'$ and $q \to_{\mathsf{e}} q'$: then, $u \to_{\mathsf{e}} r'q' \ {}_{\mathsf{e}}\!\leftarrow s$;
- *Application Left for both* $t \to_{\mathsf{e}} u$ *and* $t \to_{\mathsf{e}} s$, *i.e.* $t \coloneqq rq \to_{\mathsf{e}} r'q =: u$ and $t \to_{\mathsf{e}} r''q =: s$ with $r' \ {}_{\mathsf{e}}\!\leftarrow r \to_{\mathsf{e}} r''$: by *i.h.*, there exists $r_0 \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{e}} r_0 \ {}_{\mathsf{e}}\!\leftarrow r''$, hence $u \to_{\mathsf{e}} r_0 q \ {}_{\mathsf{e}}\!\leftarrow s$;
- *Application Right for both* $t \to_{\mathsf{e}} u$ *and* $t \to_{\mathsf{e}} s$, *i.e.* $t \coloneqq qr \to_{\mathsf{e}} qr' =: u$ and $t \to_{\mathsf{e}} qr'' =: s$ with $r' \ {}_{\mathsf{e}}\!\leftarrow r \to_{\mathsf{e}} r''$: by *i.h.*, there exists $r_0 \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{e}} r_0 \ {}_{\mathsf{e}}\!\leftarrow r''$, hence $u \to_{\mathsf{e}} qr_0 \ {}_{\mathsf{e}}\!\leftarrow s$;
- *ES Left for* $t \to_{\mathsf{e}} u$ and *ES Right for* $t \to_{\mathsf{e}} s$, *i.e.* $t \coloneqq r[x{\leftarrow}q] \to_{\mathsf{e}} r'[x{\leftarrow}q] =: u$ and $t \to_{\mathsf{e}} r[x{\leftarrow}q'] =: s$ with $r \to_{\mathsf{e}} r'$ and $q \to_{\mathsf{e}} q'$: then, $u \to_{\mathsf{e}} r'[x{\leftarrow}q'] \ {}_{\mathsf{e}}\!\leftarrow s$;
- *ES Left for both* $t \to_{\mathsf{e}} u$ *and* $t \to_{\mathsf{e}} s$, *i.e.* $t \coloneqq r[x{\leftarrow}q] \to_{\mathsf{e}} r'[x{\leftarrow}q] =: u$ and $t \to_{\mathsf{e}} r''[x{\leftarrow}q] =: s$ with $r' \ {}_{\mathsf{e}}\!\leftarrow r \to_{\mathsf{e}} r''$: by *i.h.*, there exists $r_0 \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{e}} r_0 \ {}_{\mathsf{e}}\!\leftarrow r''$, hence $u \to_{\mathsf{e}} r_0[x{\leftarrow}q] \ {}_{\mathsf{e}}\!\leftarrow s$;
- *ES Right for both* $t \to_{\mathsf{e}} u$ *and* $t \to_{\mathsf{e}} s$, *i.e.* $t \coloneqq q[x{\leftarrow}r] \to_{\mathsf{e}} q[x{\leftarrow}r'] =: u$ and $t \to_{\mathsf{e}} q[x{\leftarrow}r''] =: s$ with $r' \ {}_{\mathsf{e}}\!\leftarrow r \to_{\mathsf{e}} r''$: by *i.h.*, there exists $r_0 \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{e}} r_0 \ {}_{\mathsf{e}}\!\leftarrow r''$, hence $u \to_{\mathsf{e}} q[x{\leftarrow}r_0] \ {}_{\mathsf{e}}\!\leftarrow s$.

Note that in [3, Lemma 11] it has just been proved the strong confluence of $\to_{\mathsf{vsub}}$, not of $\to_{\mathsf{m}}$ or $\to_{\mathsf{e}}$.

3. We show that $\to_{\mathsf{e}}$ and $\to_{\mathsf{m}}$ strongly commute, *i.e.* if $u \ {}_{\mathsf{e}}\!\leftarrow t \to_{\mathsf{m}} s$, then $u \neq s$ and there is $t' \in \Lambda_{\mathsf{vsub}}$ such that $u \to_{\mathsf{m}} t' \ {}_{\mathsf{e}}\!\leftarrow s$. The proof is by induction on the definition of $t \to_{\mathsf{e}} u$. The proof that $u \neq s$ is left to the reader. Since the $\to_{\mathsf{e}}$ and $\to_{\mathsf{m}}$ cannot reduce under $\lambda$'s, all $\mathsf{vsub}$-values are $\mathsf{m}$-normal and $\mathsf{e}$-normal. So, there are the following cases.
   - *Step at the Root for $t \to_{\mathsf{e}} u$* and *ES Left for $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq r[z{\leftarrow}L\langle v\rangle] \to_{\mathsf{e}} L\langle r\{z{\leftarrow}v\}\rangle \eqqcolon u$ and $t \to_{\mathsf{m}} r'[z{\leftarrow}L\langle v\rangle] \eqqcolon s$ with $r \to_{\mathsf{m}} r'$: then $u \to_{\mathsf{m}} L\langle r'\{z{\leftarrow}v\}\rangle \ {}_{\mathsf{e}}\!\leftarrow u$;
   - *Step at the Root for $t \to_{\mathsf{e}} u$* and *ES Right for $t \to_{\mathsf{m}} s$, i.e.*
   $$\begin{aligned} t &\coloneqq r[z{\leftarrow}v[x_1{\leftarrow}t_1]\dots[x_n{\leftarrow}t_n]] \\ &\to_{\mathsf{e}} r\{z{\leftarrow}v\}[x_1{\leftarrow}t_1]\dots[x_n{\leftarrow}t_n] \eqqcolon u \end{aligned}$$
   and $t \to_{\mathsf{m}} r[z{\leftarrow}v[x_1{\leftarrow}t_1]\dots[x_j{\leftarrow}t'_j]\dots[x_n{\leftarrow}t_n]] \eqqcolon s$ for some $n > 0$, and $t_j \to_{\mathsf{m}} t'_j$ for some $1 \leq j \leq n$: then, $u \to_{\mathsf{m}} r\{z{\leftarrow}v\}[x_1{\leftarrow}t_1]\dots[x_j{\leftarrow}t'_j]\dots[x_n{\leftarrow}t_n] \ {}_{\mathsf{e}}\!\leftarrow s$;
   - *Application Left for $t \to_{\mathsf{e}} u$* and *Application Right for $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq rq \to_{\mathsf{e}} r'q \eqqcolon u$ and $t \to_{\mathsf{m}} rq' \eqqcolon s$ with $r \to_{\mathsf{e}} r'$ and $q \to_{\mathsf{m}} q'$: then, $t \to_{\mathsf{m}} r'q' \ {}_{\mathsf{e}}\!\leftarrow u$;
   - *Application Left for both $t \to_{\mathsf{e}} u$ and $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq rq \to_{\mathsf{e}} r'q \eqqcolon u$ and $t \to_{\mathsf{m}} r''q \eqqcolon s$ with $r' \ {}_{\mathsf{e}}\!\leftarrow r \to_{\mathsf{m}} r''$: by *i.h.*, there exists $p \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{m}} p \ {}_{\mathsf{e}}\!\leftarrow r''$, hence $u \to_{\mathsf{m}} pq \ {}_{\mathsf{e}}\!\leftarrow s$;
   - *Application Left for $t \to_{\mathsf{e}} u$* and *Step at the Root for $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq (\lambda x.q)[x_1{\leftarrow}t_1]\dots[x_n{\leftarrow}t_n]r \to_{\mathsf{e}} (\lambda x.q)[x_1{\leftarrow}t_1]\dots[x_j{\leftarrow}t'_j]\dots[x_n{\leftarrow}t_n]r \eqqcolon u$ with $n > 0$ and $t_j \to_{\mathsf{e}} t'_j$ for some $1 \leq j \leq n$, and $t \to_{\mathsf{m}} q[x{\leftarrow}r][x_1{\leftarrow}t_1]\dots[x_n{\leftarrow}t_n] \eqqcolon s$: then,
   $$u \to_{\mathsf{m}} q[x{\leftarrow}r][x_1{\leftarrow}t_1]\dots[x_j{\leftarrow}t'_j]\dots[x_n{\leftarrow}t_n] \ {}_{\mathsf{e}}\!\leftarrow s;$$
   - *Application Right for $t \to_{\mathsf{e}} u$* and *Application Left for $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq qr \to_{\mathsf{e}} qr' \eqqcolon u$ and $t \to_{\mathsf{m}} q'r \eqqcolon s$ with $r \to_{\mathsf{e}} r'$ and $q \to_{\mathsf{m}} q'$: then, $u \to_{\mathsf{m}} q'r' \ {}_{\mathsf{e}}\!\leftarrow s$;
   - *Application Right for both $t \to_{\mathsf{e}} u$ and $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq qr \to_{\mathsf{e}} qr' \eqqcolon u$ and $t \to_{\mathsf{m}} qr'' \eqqcolon s$ with $r' \ {}_{\mathsf{e}}\!\leftarrow r \to_{\mathsf{m}} r''$: by *i.h.*, there exists $p \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{m}} p \ {}_{\mathsf{e}}\!\leftarrow r''$, hence $u \to_{\mathsf{m}} qp \ {}_{\mathsf{e}}\!\leftarrow s$;
   - *Application Right for $t \to_{\mathsf{e}} u$* and *Step at the Root for $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq L\langle\lambda x.q\rangle r \to_{\mathsf{e}} L\langle\lambda x.q\rangle r' \eqqcolon u$ with $r \to_{\mathsf{e}} r'$, and $t \to_{\mathsf{m}} L\langle q[x{\leftarrow}r]\rangle \eqqcolon s$: then, $u \to_{\mathsf{m}} L\langle q[x{\leftarrow}r']\rangle \ {}_{\mathsf{e}}\!\leftarrow s$;
   - *ES Left for $t \to_{\mathsf{e}} u$* and *ES Right for $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq r[x{\leftarrow}q] \to_{\mathsf{e}} r'[x{\leftarrow}q] \eqqcolon u$ and $t \to_{\mathsf{m}} r[x{\leftarrow}q'] \eqqcolon s$ with $r \to_{\mathsf{e}} r'$ and $q \to_{\mathsf{m}} q'$: then, $u \to_{\mathsf{m}} r'[x{\leftarrow}q'] \ {}_{\mathsf{e}}\!\leftarrow s$;
   - *ES Left for both $t \to_{\mathsf{e}} u$ and $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq r[x{\leftarrow}q] \to_{\mathsf{e}} r'[x{\leftarrow}q] \eqqcolon u$ and $t \to_{\mathsf{m}} r''[x{\leftarrow}q] \eqqcolon s$ with $r' \ {}_{\mathsf{e}}\!\leftarrow r \to_{\mathsf{m}} r''$: by *i.h.*,there exists $p \in \Lambda_{\mathsf{vsub}}$ such that $r' \to_{\mathsf{m}} p \ {}_{\mathsf{e}}\!\leftarrow r''$, hence $u \to_{\mathsf{m}} p[x{\leftarrow}q] \ {}_{\mathsf{e}}\!\leftarrow s$;
   - *ES Right for $t \to_{\mathsf{e}} u$* and *ES Left for $t \to_{\mathsf{m}} s$, i.e.* $t \coloneqq q[x{\leftarrow}r] \to_{\mathsf{e}} q[x{\leftarrow}r'] \eqqcolon u$ and $t \to_{\mathsf{m}} q'[x{\leftarrow}r] \eqqcolon s$ with $r \to_{\mathsf{e}} r'$ and $q \to_{\mathsf{m}} q'$: then, $u \to_{\mathsf{m}} q'[x{\leftarrow}r'] \ {}_{\mathsf{e}}\!\leftarrow s$;

   - *ES Right for both* $t \to_{\mathsf{e}} u$ *and* $t \to_{\mathsf{m}} s$*, i.e.* $t \coloneqq q[x{\leftarrow}r] \to_{\mathsf{e}} q[x{\leftarrow}r'] \eqqcolon u$ and $t \to_{\mathsf{m}} q[x{\leftarrow}r''] \eqqcolon s$ with $r \;{}_{\mathsf{e}}{\leftarrow}\; r' \to_{\mathsf{m}} r''$: by *i.h.*, there exists $p \in \Lambda_{\mathtt{vsub}}$ such that $r \to_{\mathsf{m}} p \;{}_{\mathsf{e}}{\leftarrow}\; r''$, hence $u \to_{\mathsf{m}} q[x{\leftarrow}p] \;{}_{\mathsf{e}}{\leftarrow}\; s$.

4. It follows immediately from strong confluence of $\to_{\mathsf{m}}$ and $\to_{\mathsf{e}}$ (Prop. 4.1), strong commutation of $\to_{\mathsf{m}}$ and $\to_{\mathsf{e}}$ (Prop. 4.2) and Hindley-Rosen (Lemma 26). A different proof of the strong confluence of $\to_{\mathtt{vsub}}$ (without information about the number of steps) is in [3, Lemma 11].
5. The intuition behind the proof is that any $\mathsf{m}$-step creates a new ES, any $\mathsf{e}$-step erases an ES. Formally, let $u \in \Lambda_{\mathtt{vsub}}$ such that $d\colon t \to_{\mathtt{vsub}}^* u$. We prove by induction on $|d|_{\mathtt{vsub}} \in \mathbb{N}$ that $|d|_{\mathsf{e}} = |d|_{\mathsf{m}} - |u|_{\mathrm{ES}}$ (where $|u|_{\mathrm{ES}}$ is the number of ES in $u$) and any $\mathtt{vsub}$-value that is a subterm of $u$ is a value (without ES).
   If $|d|_{\mathtt{vsub}} = 0$, then $u = t \in \Lambda$, then we can conclude.
   Suppose $|d|_{\mathtt{vsub}} > 0$: then, $d$ is the concatenation of $d'\colon t \to_{\mathtt{vsub}}^* s$ and $s \to_{\mathtt{vsub}} u$, for some $s \in \Lambda_{\mathtt{vsub}}$. By *i.h.*, $|d'|_{\mathsf{e}} = |d'|_{\mathsf{m}} - |s|_{\mathrm{ES}}$ and that every $\mathtt{vsub}$-value that is a subterm of $s$ is a value (without ES). There are two cases:
   - $s \coloneqq E\langle r[x{\leftarrow}L\langle v\rangle]\rangle \to_{\mathsf{e}} E\langle L\langle r\{x{\leftarrow}v\}\rangle\rangle \eqqcolon u$, then $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}}$ and $|s|_{\mathrm{ES}} = |u|_{\mathrm{ES}} + 1$, since $|v|_{\mathrm{ES}} = 0$ by *i.h.*; therefore $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}} + 1 = |d'|_{\mathsf{m}} - |s|_{\mathrm{ES}} + 1 = |d|_{\mathsf{m}} - |u|_{\mathrm{ES}}$ and any $\mathtt{vsub}$-value that is a subterm of $u$ is a value (without ES).
   - $s \coloneqq E\langle L\langle \lambda x.r\rangle q\rangle \to_{\mathsf{m}} E\langle L\langle r[x{\leftarrow}q]\rangle\rangle \eqqcolon u$, then $|u|_{\mathrm{ES}} = |s|_{\mathrm{ES}} + 1$ and $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}} + 1$, therefore $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}} = |d'|_{\mathsf{m}} - |s|_{\mathrm{ES}} = |d|_{\mathsf{m}} - |u|_{\mathrm{ES}}$. Moreover, the new occurrence of ES $[x{\leftarrow}q]$ in $u$ cannot be under the scope of a $\lambda$, otherwise the redex in $s$ which is fired in the $\mathsf{m}$-step would be under the scope of a $\lambda$, but this is impossible since $\to_{\mathsf{m}}$ is a weak reduction. So, any $\mathtt{vsub}$-value that is a subterm of $u$ is a value (without ES).

□

*Open CBV 3: the Shuffling Calculus* $\lambda_{\mathtt{sh}}$

**Definition 30 (Occurrences).** *For all* $t \in \Lambda$*, let* $[t]_\lambda$ *be the number of occurrences of* $\lambda$ *in* $t$*, and* $[t]_x$ *be the number of free occurrences of the variable* $x$ *in* $t$*, and* $\mathsf{sub}_u(t)$ *be the number of occurrences in* $t$ *of the term* $u$*.*

*Remark 31.* Since $\to_{\beta_v^\flat}$ and $\to_{\sigma^\flat}$ do not reduce under $\lambda$'s without argument, every value is $\beta_v^\flat$-normal and $\sigma^\flat$-normal, and hence $\mathtt{sh}$-normal.

*Remark 32.* The reduction $\to_{\sigma^\flat}$ is just the closure under balanced contexts of the binary relation $\mapsto_\sigma = \mapsto_{\sigma_1} \cup \mapsto_{\sigma_3}$ on $\Lambda$ (see definitions in Fig. 4).

**Lemma 33.** *Let* $t, t' \in \Lambda$*.*

1. *For every value* $v$*, if* $t \to_{\sigma^\flat} t'$ *then* $(\lambda x.t')v \neq t\{x{\leftarrow}v\}$*.*
2. *If* $t \to_{\sigma^\flat} t'$ *then* $t \neq t'$*.*
3. *For every value* $v$*, one has* $t\{x{\leftarrow}v\} \neq \lambda x.tv$*.*

*Proof.*

1. By induction on the definition of $t \to_{\sigma^\flat} t'$, using Remark 32.
2. In [8, Proposition 2] it has been proved that there exists a size $\#\colon \Lambda \to \mathbb{N}$ such that if $t \to_\sigma t'$ then $\#(t) > \#(t')$, where $\to_\sigma$ is just the extension of $\to_{\sigma^\flat}$ obtained by allowing reductions under $\lambda$'s. Therefore, $\to_{\sigma^\flat} \subseteq \to_\sigma$ and hence if $t \to_{\sigma^\flat} t'$ then $\#(t) > \#(t')$, in particular $t \neq t'$.
3. According to Definition 30, $[t\{x{\leftarrow}v\}]_\lambda = [t]_\lambda + [v]_\lambda \cdot [t]_x$ and $[\lambda x.tv]_\lambda = 1 + [t]_\lambda + [v]_\lambda$, and $[t\{x{\leftarrow}v\}]_x = [t]_x \cdot [v]_x$ and $[\lambda x.tv]_x = 0$.
   Suppose $t\{x{\leftarrow}v\} = \lambda x.tv$: then, $[t\{x{\leftarrow}v\}]_\lambda = [\lambda x.tv]_\lambda$ and $[t\{x{\leftarrow}v\}]_x = [\lambda x.tv]_x$, thus
   $$[v]_\lambda \cdot [t]_x = 1 + [v]_\lambda \qquad\qquad [t]_x \cdot [v]_x = 0. \tag{3}$$
   The only solution to the first equation of (3) is $[v]_\lambda = 1$ and $[t]_x = 2$, whence $[v]_x = 0$ according to the second equation of (3). As $x \notin \mathtt{fv}(v)$, one has $\mathsf{sub}_v(\lambda x.tv) = 1 + \mathsf{sub}_v(t)$ and $\mathsf{sub}_v(t\{x{\leftarrow}v\}) = \mathsf{sub}_v(t) + [t]_x = \mathsf{sub}_v(t) + 2$, therefore $\mathsf{sub}_v(\lambda x.tv) \neq \mathsf{sub}_v(t\{x{\leftarrow}v\})$ and hence $\lambda x.tv \neq t\{x{\leftarrow}v\}$. Contradiction. □

**Proposition 6** (Basic Properties of $\lambda_{\mathtt{sh}}$, [8]). See p. 10

1. *Let $t, u, s \in \Lambda$: if $t \to_{\beta_v^\flat} u$ and $t \to_{\sigma^\flat} s$ then $u \neq s$.*
2. *$\to_{\sigma^\flat}$ is strongly normalizing and (not strongly) confluent.*
3. *$\to_{\mathtt{sh}}$ is (not strongly) confluent.*
4. *Let $t \in \Lambda$: $t$ is strongly $\mathtt{sh}$-normalizable iff $t$ is $\mathtt{sh}$-normalizable.*

*Proof.*

1. By induction on $t \in \Lambda$. According to the definition of $t \to_{\sigma^\flat} s$ and Remark 32, the following cases are impossible.
   - *Step at the root for* $t \to_{\beta_v^\flat} u$ and either the *Step at the root* or the *Application Left* or the *Application Right for* $t \to_{\sigma^\flat} s$. Indeed, if $t = (\lambda x.r)v \mapsto_{\beta_v} r\{x{\leftarrow}v\} = u$ then $\lambda x.r$ and $v$ are $\sigma^\flat$-normal by Remark 31; moreover $t$ is neither a $\sigma_1$-redex nor a $\sigma_3$-redex, because $\lambda x.r$ and $v$, respectively, are not applications.
   - *Application Left for* $t \to_{\beta_v^\flat} u$ and *Step inside a $\beta$-context for* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\beta_v^\flat} pq = u$ with $r \to_{\beta_v^\flat} p$, and $t = (\lambda x.r')q \to_{\sigma^\flat} (\lambda x.m)q = s$ with $r = \lambda x.r'$ and $r' \to_{\sigma^\flat} m$. Indeed $r$ is $\beta_v^\flat$-normal by Remark 31.
   - *Step inside a $\beta$-context for* $t \to_{\beta_v^\flat} u$ and *Application Left for* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\sigma^\flat} pq = s$ with $r \to_{\sigma^\flat} p$, and $t = (\lambda x.r')q \to_{\beta_v^\flat} (\lambda x.m)q = u$ with $r = \lambda x.r'$ and $r' \to_{\beta_v^\flat} m$. Indeed $r$ is $\sigma^\flat$-normal by Remark 31.

   Therefore, according to the definition of $t \to_{\sigma^\flat} s$ and Remark 32, there are "only" eleven cases.
   - *Step at the root for* $t \to_{\beta_v^\flat} u$ and *Step inside a $\beta$-context for* $t \to_{\sigma^\flat} s$, *i.e.* $t = (\lambda x.r)v \mapsto_{\beta_v} r\{x{\leftarrow}v\} = u$ and $t = (\lambda x.r)v \to_{\sigma^\flat} (\lambda x.r')v = s$ with $r \to_{\sigma^\flat} r'$. By Lemma 33.1, $u \neq s$.

- *Application Left for* $t \to_{\beta_v^\flat} u$ and *Step at the root for* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\beta_v^\flat} pq = u$ with $r \to_{\beta_v^\flat} p$, and $t \mapsto_\sigma s$ (see Remark 32). It is impossible that $t \mapsto_{\sigma_3} s$, otherwise $r$ would be a value and hence $\beta_v^\flat$-normal by Remark 31, but this contradicts that $r \to_{\beta_v^\flat} p$. Thus, $t = (\lambda x.r')r''q \mapsto_{\sigma_1} (\lambda x.r'q)r'' = s$ with $x \notin \mathtt{fv}(q)$ and $r = (\lambda x.r')r''$. We claim that $u \neq s$. Indeed, if $u = s$ then $q = r''$ and $p = \lambda x.r'q$ with $r = (\lambda x.r')q \to_{\beta_v^\flat} \lambda x.r'q = p$, hence necessarily $r \mapsto_{\beta_v} p$ (*i.e.* $r \to_{\beta_v^\flat} p$ by a step at the root) and thus $q$ is a value and $\lambda x.r'q = p = r'\{x{\leftarrow}q\}$, but this is impossible by Lemma 33.3.
- *Application Left for* $t \to_{\beta_v^\flat} u$ *and* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\beta_v^\flat} pq = u$ and $t = rq \to_{\sigma^\flat} mq = s$ with $r \to_{\beta_v^\flat} p$ and $r \to_{\sigma^\flat} m$. By *i.h.*, $p \neq m$ and hence $u = pq \neq mq = s$.
- *Application Left for* $t \to_{\beta_v^\flat} u$ and *Application Right for* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\beta_v^\flat} pq = u$ and $t = rq \to_{\sigma^\flat} rm = s$, with $r \to_{\beta_v^\flat} p$ and $q \to_{\sigma^\flat} m$. By Lemma 33.2, $q \neq m$ and hence $u = pq \neq rm = s$.
- *Application Right for* $t \to_{\beta_v^\flat} u$ and *Step at the root for* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\beta_v^\flat} rp = u$ with $q \to_{\beta_v^\flat} p$, and $t \mapsto_\sigma s$ (see Remark 32).
  - If $t \mapsto_{\sigma_1} s$ then $t = (\lambda x.r')r''q \mapsto_{\sigma_1} (\lambda x.r'q)r'' = s$ with $x \notin \mathtt{fv}(q)$ and $r = (\lambda x.r')r''$. We claim that $u \neq s$. Indeed, if $u = s$ then $p = r''$ and $r = \lambda x.r'q$, therefore $(\lambda x.r')p = r = \lambda x.r'q$ which is impossible.
  - If $t \mapsto_{\sigma_3} s$ then $t = r((\lambda x.q')q'') \mapsto_{\sigma_3} (\lambda x.rq')q'' = s$ where $r$ is a value, $x \notin \mathtt{fv}(r)$ and $q = (\lambda x.q')q''$. We claim that $u \neq s$. Indeed, if $u = s$ then $r = \lambda x.rq'$ which is impossible.
- *Application Right for* $t \to_{\beta_v^\flat} u$ *and* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\beta_v^\flat} pq = u$ and $t = rq \to_{\sigma^\flat} mq = s$ with $q \to_{\beta_v^\flat} p$ and $q \to_{\sigma^\flat} m$. By *i.h.*, $p \neq m$ and hence $u = rp \neq rm = s$.
- *Application Right for* $t \to_{\beta_v^\flat} u$ and *Application Left for* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\beta_v^\flat} rp = u$ and $t = rq \to_{\sigma^\flat} mq = s$, with $q \to_{\beta_v^\flat} p$ and $r \to_{\sigma^\flat} m$. By Lemma 33.2, $r \neq m$ and hence $u = rp \neq mq = s$.
- *Application Right for* $t \to_{\beta_v^\flat} u$ and *Step inside a* $\beta$*-context for* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\beta_v^\flat} rp = u$ with $q \to_{\beta_v^\flat} p$, and $t = (\lambda x.r')q \to_{\sigma^\flat} (\lambda x.m)q = s$ with $r = \lambda x.r'$ and $r' \to_{\sigma^\flat} m$. By Lemma 33.2, $r' \neq m$ whence $r = \lambda x.r' \neq \lambda x.m$ and thus $u \neq s$.
- *Step inside a* $\beta$*-context for* $t \to_{\beta_v^\flat} u$ and *Step at the root for* $t \to_{\sigma^\flat} s$, *i.e.* $t = (\lambda x.r)q \to_{\beta_v^\flat} (\lambda x.r')q = u$ with $r \to_{\beta_v^\flat} r'$, and $t \mapsto_\sigma s$ (see Remark 32). It is impossible that $t = (\lambda x.r)q \mapsto_{\sigma_1} s$ because $\lambda x.r$ is not an application. Thus, $t = (\lambda x.r)((\lambda y.q')q'') \mapsto_{\sigma_3} (\lambda y.(\lambda x.r)q')q'' = s$ with $q = (\lambda y.q')q''$ and $y \notin \mathtt{fv}(\lambda x.r)$, therefore $q \neq q''$ and hence $u \neq s$.
- *Step inside a* $\beta$*-context for* $t \to_{\beta_v^\flat} u$ and *Application Right for* $t \to_{\sigma^\flat} s$, *i.e.* $t = rq \to_{\sigma^\flat} rp = s$ with $q \to_{\sigma^\flat} p$, and $t = (\lambda x.r')q \to_{\beta_v^\flat} (\lambda x.m)q = u$ with $r = \lambda x.r'$ and $r' \to_{\beta_v^\flat} m$. By Lemma 33.2, $q \neq p$ whence $u \neq s$.
- *Step inside a* $\beta$*-context for* $t \to_{\beta_v^\flat} u$ *and* $t \to_{\sigma^\flat} s$, *i.e.* $t = (\lambda x.r)q \to_{\beta_v} (\lambda x.p)q = u$ and $t = (\lambda x.r)q \to_{\sigma^\flat} (\lambda x.m)q = s$ with $r \to_{\beta_v^\flat} p$ and $r \to_{\sigma^\flat} m$. By *i.h.*, $p \neq m$ and hence $u \neq s$.

2. In [8, Proposition 2] it has been proved that $\to_\sigma$ is strongly normalizing, where $\to_\sigma$ is just the extension of $\to_{\sigma^\flat}$ obtained by allowing reductions under $\lambda$'s. Therefore, $\to_{\sigma^\flat} \subseteq \to_\sigma$ and hence $\to_{\sigma^\flat}$ is strongly normalizing.

The (not strong) confluence of $\to_{\sigma^\flat}$ has been proved in [8, Lemma 9.ii], where $\to_{\sigma^\flat}$ is denoted by $\to_{\mathtt{w}[\sigma]}$.
3. See [8, Proposition 10], where $\to_{\mathtt{sh}}$ is denoted by $\to_{\mathtt{w}}$.
4. See [8, Theorem 24], where $\to_{\mathtt{sh}}$ is denoted by $\to_{\mathtt{w}}$. □

*Open CBV 4: the Value Sequent Calculus* $\lambda_{\mathtt{vseq}}$. We aim to prove the strong confluence of $\to_{\mathtt{vseq}}$.

Note that values are closed under substitution: for all values $v, v'$, $v\{x{\leftarrow}v'\}$ is a value. Moreover, values are $\bar{\lambda}$-, $\tilde{\mu}$- and $\mathtt{vseq}$-normal.

**Definition 34.** *For any* $\mathsf{r} \in \{\bar{\lambda}, \tilde{\mu}, \mathtt{vseq}\}$*, given two environments* $e$ *and* $e'$*, we define* $e \to_{\mathsf{r}} e'$ *by induction on* $e$*. If* $e = \epsilon$ *then there is no* $e'$ *such that* $e \to_{\mathsf{r}} e'$*. If* $e = \tilde{\mu}x.c$ *then* $e' = \tilde{\mu}x.c'$ *and* $c \to_{\mathsf{r}} c'$*. If* $e = v{\cdot}e_0$ *then* $e' = v{\cdot}e_0'$ *and* $e_0 \to_{\mathsf{r}} e_0'$*.*

*Remark 35.* Let $c$ and $c'$ be commands and $\mathsf{r} \in \{\bar{\lambda}, \tilde{\mu}, \mathtt{vseq}\}$. One has $c \to_{\mathsf{r}} c'$ iff

- either $c = \langle \lambda x.c_0 \,|\, v{\cdot}e \rangle$ and $c' = \langle v \,|\, (\tilde{\mu}x.c_0)@e \rangle$,
- or $c = \langle v \,|\, \tilde{\mu}x.c_0 \rangle$ and $c' = c_0\{x{\leftarrow}v\}$,
- or $c = \langle v \,|\, e \rangle$, $c' = \langle v \,|\, e' \rangle$ and $e \to_{\mathsf{r}} e'$.

**Lemma 36 (Substitution).** *Let* $c$ *and* $c'$ *be commands,* $e$ *and* $e'$ *be environments,* $v$ *be a value and* $x$ *be a variable. Let* $\mathsf{r} \in \{\bar{\lambda}, \tilde{\mu}, \mathtt{vseq}\}$*.*
1. *If* $e \to_{\mathsf{r}} e'$ *then* $e\{x{\leftarrow}v\} \to_{\mathsf{r}} e'\{x{\leftarrow}v\}$*;*
2. *If* $c \to_{\mathsf{r}} c'$ *then* $c\{x{\leftarrow}v\} \to_{\mathsf{r}} c'\{x{\leftarrow}v\}$*.*

*Proof.* Both points are proved simultaneously by mutual induction on $c$ and $e$. Cases:

1. *Step at the root for* $c \to_{\bar{\lambda}} c'$*, i.e.* $c := \langle \lambda y.c_0 \,|\, v'{\cdot}e_0 \rangle \mapsto_{\bar{\lambda}} \langle v' \,|\, (\tilde{\mu}y.c_0)@e_0 \rangle =: c'$. We can suppose without loss of generality that $y \notin \mathtt{fv}(v) \cup \{x\}$. So, $c\{x{\leftarrow}v\} = \langle \lambda y.c_0\{x{\leftarrow}v\} \,|\, v'\{x{\leftarrow}v\}{\cdot}e_0\{x{\leftarrow}v\} \rangle \to_{\bar{\lambda}} \langle v'\{x{\leftarrow}v\} \,|\, (\tilde{\mu}y.c_0\{x{\leftarrow}v\})@e_0\{x{\leftarrow}v\} \rangle = c'\{x{\leftarrow}v\}$.
2. *Step at the root for* $c \to_{\tilde{\mu}} c'$*, i.e.* $c := \langle v' \mid \tilde{\mu}y.c_0 \rangle \mapsto_{\tilde{\mu}} c_0\{y{\leftarrow}v'\} =: c'$. We can suppose without loss of generality that $y \notin \mathtt{fv}(v) \cup \{x\}$. So, $c\{x{\leftarrow}v\} = \langle v'\{x{\leftarrow}v\} \,|\, \tilde{\mu}y.c_0\{x{\leftarrow}v\} \rangle \to_{\tilde{\mu}} c_0\{x{\leftarrow}v\}\{y{\leftarrow}v'\{x{\leftarrow}v\}\} = c'\{x{\leftarrow}v\}$.
3. *Environment step for* $c \to_{\mathsf{r}} c'$*, i.e.* $c := \langle v' \,|\, e \rangle \to_{\mathsf{r}} \langle v' \,|\, e' \rangle =: c'$ with $e \to_{\mathsf{r}} e'$: by *i.h.*, $e\{x{\leftarrow}v\} \to_{\mathsf{r}} e'\{x{\leftarrow}v\}$ and hence $c\{x{\leftarrow}v\} = \langle v'\{x{\leftarrow}v\} \,|\, e\{x{\leftarrow}v\} \rangle \to_{\mathsf{r}} \langle v'\{x{\leftarrow}v\} \,|\, e'\{x{\leftarrow}v\} \rangle = c'\{x{\leftarrow}v\}$ according to Remark 35.
4. *$\tilde{\mu}$-environment step for* $e \to_{\mathsf{r}} e'$*, i.e.* $e := \tilde{\mu}y.c \to_{\mathsf{r}} \tilde{\mu}y.c' =: e'$ with $c \to_{\mathsf{r}} c'$. We can suppose without loss of generality that $y \notin \mathtt{fv}(v) \cup \{x\}$. By *i.h.* $c\{x{\leftarrow}v\} \to_{\mathsf{r}} c'\{x{\leftarrow}v\}$, and thus $e\{x{\leftarrow}v\} = \tilde{\mu}y.c\{x{\leftarrow}v\} \to_{\mathsf{r}} \tilde{\mu}y.c'\{x{\leftarrow}v\} = e'\{x{\leftarrow}v\}$ according to Definition 34.
5. *Environment step for* $e \to_{\mathsf{r}} e'$*, i.e.* $e := v'{\cdot}e_0 \to_{\mathsf{r}} v'{\cdot}e_0' =: e'$ with $e_0 \to_{\mathsf{r}} e_0'$: by *i.h.*, $e_0\{x{\leftarrow}v\} \to_{\mathsf{r}} e_0'\{x{\leftarrow}v\}$ and hence $e\{x{\leftarrow}v\} = v'\{x{\leftarrow}v\}{\cdot}e_0\{x{\leftarrow}v\} \to_{\mathsf{r}} v'\{x{\leftarrow}v\}{\cdot}e_0'\{x{\leftarrow}v\} = e'\{x{\leftarrow}v\}$ according to Definition 34. □

**Lemma 37 (Append).** *Let* $\mathsf{r} \in \{\bar{\lambda}, \tilde{\mu}, \mathtt{vseq}\}$*,* $c$ *be a command and* $e_0$*,* $e$ *and* $e'$ *be environments. If* $e \to_{\mathsf{r}} e'$ *then* $e_0@e \to_{\mathsf{r}} e_0@e'$ *and* $c@e \to_{\mathsf{r}} c@e'$*.*

*Proof.* We prove simultaneously that $e_0@e \to_{\mathsf{r}} e_0@e'$ and $c@e \to_{\mathsf{r}} c@e'$ by mutual induction on $c$ and $e_0$. Cases:

1. $c = \langle v \,|\, e_0 \rangle$: by *i.h.*, $e_0@e \to_{\mathsf{r}} e_0@e'$. Thus, $c@e = \langle v \,|\, e_0@e \rangle \to_{\mathsf{r}} \langle v \,|\, e_0@e' \rangle = c@e'$ according to Remark 35.
2. $e_0 = \epsilon$: then, $e_0@e = e \to_{\mathsf{r}} e' = e_0@e'$.
3. $e_0 = v_0 \cdot e_0'$: by *i.h.*, $e_0'@e \to_{\mathsf{r}} e_0'@e'$. Hence, $e_0@e = v \cdot (e_0'@e) \to_{\mathsf{r}} v \cdot (e_0'@e') = e_0@e'$ according to Definition 34.
4. $e_0 = \tilde{\mu}y.c$: we can suppose without loss of generality that $y \notin \mathtt{fv}(v) \cup \mathtt{fv}(e) \cup \{x\}$, whence $y \notin \mathtt{fv}(e')$. By *i.h.*, $c@e \to_{\mathsf{r}} c@e'$. Hence, $e_0@e = \tilde{\mu}y.c@e \to_{\mathsf{r}} \tilde{\mu}y.c@e' = e_0@e'$ according to Definition 34. □

**Lemma 38 (Append Commutes).**

1. Evaluation Contexts*:* $C\langle c \rangle@e = C\langle c@e \rangle$ *and* $D\langle e' \rangle@e = D\langle e'@e \rangle$*.*
2. Rewriting Steps in Commands and Environments*: if* $c \to_{\bar{\lambda}\tilde{\mu}} c'$ *(resp.* $e \to_{\bar{\lambda}\tilde{\mu}} e'$*) then* $c@e_0 \to_{\mathtt{vseq}} c'@e_0$ *(resp.* $e@e_0 \to_{\mathtt{vseq}} e'@e_0$*).*

*Proof.*

1. By mutual induction on $C$ and $D$ (see Fig. 5). Cases:
   - $C = \langle \cdot \rangle$: then, $C\langle c \rangle@e = c@e = C\langle c@e \rangle$.
   - $C = D\langle \tilde{\mu}x.C' \rangle$: we can suppose without loss of generality that $x \notin \mathtt{fv}(e)$. So, $C\langle c \rangle@e = D\langle \tilde{\mu}x.C'\langle c \rangle \rangle@e \overset{i.h.}{=} D\langle (\tilde{\mu}x.C'\langle c \rangle)@e \rangle = D\langle \tilde{\mu}x.(C'\langle c \rangle@e) \rangle \overset{i.h.}{=} D\langle \tilde{\mu}x.C'\langle c@e \rangle \rangle = C\langle c@e \rangle$, with we have applied the *i.h.* the first time to $D$, the second time to $C'$.
   - $D = \langle v \,|\, \langle \cdot \rangle \rangle$: then, $D\langle e' \rangle@e = \langle v \,|\, e' \rangle@e = \langle v \,|\, e'@e \rangle = D\langle e'@e \rangle$.
   - $D = D'\langle v \cdot \langle \cdot \rangle \rangle$: one has $D\langle e' \rangle@e = D'\langle v \cdot e' \rangle@e \overset{i.h.}{=} D'\langle (v \cdot e')@e \rangle = D'\langle v \cdot (e'@e) \rangle$.
2. By mutual induction on $c$ and $e$. According to Remark 35, there are three cases for $c \to_{\mathtt{vseq}} c'$:
   - either $c = \langle \lambda x.c_0 \,|\, v \cdot e \rangle \to_{\mathtt{vseq}} \langle v \,|\, (\tilde{\mu}x.c_0)@e \rangle = c'$: then, $c@e_0 = \langle \lambda x.c_0 \,|\, v \cdot (e@e_0) \rangle \to_{\mathtt{vseq}} \langle v \,|\, \tilde{\mu}x.c_0@(e@e_0) \rangle = \langle v \,|\, \tilde{\mu}x.(c_0@e)@e_0 \rangle = c'@e_0$, where the next-to-last identity holds by Lemma 38.1 taking $D = \langle v \,|\, \tilde{\mu}x.c_0@\langle \cdot \rangle \rangle$;
   - or $c = \langle v \,|\, \tilde{\mu}x.c_0 \rangle \to_{\mathtt{vseq}} c_0\{x{\leftarrow}v\} = c'$: we can suppose without loss of generality that $x \notin \mathtt{fv}(e_0)$; so, $c@e_0 = \langle v \,|\, \tilde{\mu}x.(c_0@e_0) \rangle \to_{\mathtt{vseq}} (c_0@e_0)\{x{\leftarrow}v\} = c'@e_0$;
   - or $c = \langle v \mid e \rangle \to_{\mathtt{vseq}} \langle v \mid e' \rangle = c'$ with $e \to_{\mathtt{vseq}} e'$: then, $c@e_0 = \langle v \mid e@e_0 \rangle \to_{\mathtt{vseq}} \langle v \,|\, e'@e_0 \rangle = c'@e_0$ by Remark 35.

   According to Definition 34, there are only two cases for $e \to_{\mathtt{vseq}} e'$:
   - either $e = \tilde{\mu}x.c$ and $e' = \tilde{\mu}x.c'$ and $c \to_{\mathtt{vseq}} c'$: we can suppose without loss of generality that $x \notin \mathtt{fv}(e_0)$; by *i.h.*, $c@e_0 \to_{\mathtt{vseq}} c'@e_0$ and hence $e@e_0 = \tilde{\mu}x.(c@e_0) \to_{\mathtt{vseq}} \tilde{\mu}x.(c'@e_0) = e'@e_0$;
   - or $e = v \cdot e_1$ and $e' = v \cdot e_1'$ with $e_1 \to_{\mathtt{vseq}} e_1'$; by *i.h.*, $e@e_1 \to_{\bar{\lambda}\tilde{\mu}} e'@e_1$ and hence $e@e_0 = v \cdot (e_1@e_0) \to_{\bar{\lambda}\tilde{\mu}} v \cdot (e_1'@e_0) = e'@e_0$. □

See p. 11

**Proposition 7** (Basic properties of $\bar{\lambda}\tilde{\mu}$)**.**

*1.* $\to_{\bar{\lambda}}$ *is strongly normalizing and strongly confluent.*
*2.* $\to_{\tilde{\mu}}$ *is strongly normalizing and strongly confluent.*
*3.* $\to_{\bar{\lambda}}$ *and* $\to_{\tilde{\mu}}$ *strongly commute.*
*4.* $\to_{\mathtt{vseq}}$ *is strongly confluent, and all* $\mathtt{vseq}$*-normalizing derivations* $d$ *from a command* $c$ *or an environment* $e$ *(if any) have the same length* $|d|_{\mathtt{vseq}}$*, the same number* $|d|_{\tilde{\mu}}$ *of* $\tilde{\mu}$*-steps, and the same number* $|d|_{\bar{\lambda}}$ *of* $\bar{\lambda}$*-steps.*

*Proof.*

1. Note that if $c \to_{\bar{\lambda}} c'$ then the number of occurrences of $\lambda$ in $c'$ is strictly less than in $c$: this is enough to prove that $\to_{\bar{\lambda}}$ is strongly normalizing.
   Concerning the strong confluence of $\to_{\bar{\lambda}}$, we prove that
   (a) (commands) if $c \to_{\bar{\lambda}} c_1$ and $c \to_{\bar{\lambda}} c_2$ with $c_1 \neq c_2$, then there exists $c'$ such that $c_1 \to_{\bar{\lambda}} c'$ and $c_2 \to_{\bar{\lambda}} c'$;
   (b) (environments) if $e \to_{\bar{\lambda}} e_1$ and $e \to_{\bar{\lambda}} e_2$ with $e_1 \neq e_2$, then there exists $c'$ such that $e_1 \to_{\bar{\lambda}} e'$ and $e_2 \to_{\bar{\lambda}} e'$.
   The proof is by mutual induction on $c$ and $e$. Cases:
   – *Step at the root for* $c \to_{\bar{\lambda}} c_1$ *and Step on a* $v$*-environment for* $c \to_{\bar{\lambda}} c_2$, *i.e.* $c := \langle \lambda x.c_0 \,|\, v{\cdot}e \rangle \to_{\bar{\lambda}} \langle v \,|\, (\tilde{\mu}x.c_0)@e \rangle =: c_1$ and $c \to_{\bar{\lambda}} \langle \lambda x.c_0 \,|\, v{\cdot}e' \rangle =: c_2$ with $e \to_{\bar{\lambda}} e'$. Then, $c_2 \to_{\bar{\lambda}} \langle v \mid (\tilde{\mu}x.c_0)@e' \rangle =: c'$. According to the co-substitution lemma (Lemma 37), $c_1 \to_{\bar{\lambda}} c'$.
   – *Step on an environment for both* $c \to_{\bar{\lambda}} c_1$ *and* $c \to_{\bar{\lambda}} c_2$, *i.e.* $c := \langle v \,|\, e \rangle \to_{\bar{\lambda}} \langle v \,|\, e_1 \rangle =: c_1$ and $c \to_{\bar{\lambda}} \langle v \,|\, e_2 \rangle =: c_2$ with $e_1 \;{}_{\bar{\lambda}}{\leftarrow}\; e \to_{\bar{\lambda}} e_2$. By *i.h.*, there is an environment $e'$ such that $e_1 \to_{\bar{\lambda}} e' \;{}_{\bar{\lambda}}{\leftarrow}\; e_2$. According to Remark 35, $c_1 \to_{\bar{\lambda}} c' \;{}_{\bar{\lambda}}{\leftarrow}\; c_2$ by taking $c' := \langle v \,|\, e' \rangle$.
   – *Step on a* $\tilde{\mu}$*-environment for both* $e \to_{\bar{\lambda}} e_1$ *and* $e \to_{\bar{\lambda}} e_2$, *i.e.* $e := \tilde{\mu}\alpha.c \to_{\bar{\lambda}} \tilde{\mu}\alpha.c_1 =: e_1$ and $e \to_{\bar{\lambda}} \tilde{\mu}\alpha.c_2 =: e_2$ with $c_1 \;{}_{\bar{\lambda}}{\leftarrow}\; c \to_{\bar{\lambda}} c_2$. By *i.h.*, there is a command $c'$ such that $c_1 \to_{\bar{\lambda}} c' \;{}_{\bar{\lambda}}{\leftarrow}\; c_2$. So, $e_1 \to_{\bar{\lambda}} e' \;{}_{\bar{\lambda}}{\leftarrow}\; e_2$ by taking $e' := \tilde{\mu}\alpha.c'$, according to Definition 34.
   – *Step on an environment for both* $e \to_{\bar{\lambda}} e_1$ *and* $e \to_{\bar{\lambda}} e_2$, *i.e.* $e := v{\cdot}e' \to_{\bar{\lambda}} v{\cdot}e'_1 =: e_1$ and $e \to_{\bar{\lambda}} v{\cdot}e'_2 =: e_2$ with $e'_1 \;{}_{\bar{\lambda}}{\leftarrow}\; e' \to_{\bar{\lambda}} e'_2$. By *i.h.*, there is an environment $e'_0$ such that $e'_1 \to_{\bar{\lambda}} e'_0 \;{}_{\bar{\lambda}}{\leftarrow}\; e'_2$. According to Remark 35, $e_1 \to_{\bar{\lambda}} e_0 \;{}_{\bar{\lambda}}{\leftarrow}\; e_2$ by taking $e_0 := v{\cdot}e'_0$.
2. The proof of strong normalization of $\to_{\tilde{\mu}}$ is in [16].
   Concerning the proof of strong confluence of $\to_{\tilde{\mu}}$, we prove that:
   (a) (commands) if $c \to_{\tilde{\mu}} c_1$ and $c \to_{\tilde{\mu}} c_2$ with $c_1 \neq c_2$, then there exists $c'$ such that $c_1 \to_{\tilde{\mu}} c'$ and $c_2 \to_{\tilde{\mu}} c'$;
   (b) (environments) if $e \to_{\tilde{\mu}} e_1$ and $e \to_{\tilde{\mu}} e_2$ with $e_1 \neq e_2$, then there exists $c'$ such that $e_1 \to_{\tilde{\mu}} e'$ and $e_2 \to_{\tilde{\mu}} e'$.
   The proof is by mutual induction on $c$ and $e$. Cases:
   – *Step at the root for* $c \to_{\tilde{\mu}} c_1$ *and Step on a* $\tilde{\mu}$*-environment for* $c \to_{\tilde{\mu}} c_2$, *i.e.* $c := \langle v \mid \tilde{\mu}x.c_0 \rangle \to_{\tilde{\mu}} c_0\{x{\leftarrow}v\} =: c_1$ and $c \to_{\tilde{\mu}} \langle v \mid \tilde{\mu}x.c'' \rangle =: c_2$ with $c_0 \to_{\tilde{\mu}} c''$. Then, $c_2 \to_{\tilde{\mu}} c''\{x{\leftarrow}v\} =: c'$. According to the substitution lemma (Lemma 36), $c_1 \to_{\tilde{\mu}} c'$.
   – *Step on an environment for both* $c \to_{\tilde{\mu}} c_1$ *and* $c \to_{\tilde{\mu}} c_2$, *i.e.* $c := \langle v \,|\, e \rangle \to_{\tilde{\mu}} \langle v \,|\, e_1 \rangle =: c_1$ and $c \to_{\tilde{\mu}} \langle v \,|\, e_2 \rangle =: c_2$ with $e_1 \;{}_{\tilde{\mu}}{\leftarrow}\; e \to_{\tilde{\mu}} e_2$. By *i.h.*, there is an environment $e'$ such that $e_1 \to_{\tilde{\mu}} e' \;{}_{\tilde{\mu}}{\leftarrow}\; e_2$. According to Remark 35, $c_1 \to_{\tilde{\mu}} c' \;{}_{\tilde{\mu}}{\leftarrow}\; c_2$ by taking $c' := \langle v \,|\, e' \rangle$.

- *Step on a $\tilde{\mu}$-environment for both $e \to_{\tilde{\mu}} e_1$ and $e \to_{\tilde{\mu}} e_2$, i.e. $e \coloneqq \tilde{\mu}\alpha.c \to_{\tilde{\mu}} \tilde{\mu}\alpha.c_1 =: e_1$* and $e \to_{\tilde{\mu}} \tilde{\mu}\alpha.c_2 =: e_2$ with $c_1 \ {}_{\tilde{\mu}}\!\leftarrow c \to_{\tilde{\mu}} c_2$. By *i.h.*, there is a command $c'$ such that $c_1 \to_{\tilde{\mu}} c' \ {}_{\tilde{\mu}}\!\leftarrow c_2$. So, $e_1 \to_{\tilde{\mu}} e' \ {}_{\tilde{\mu}}\!\leftarrow e_2$ by taking $e' \coloneqq \tilde{\mu}\alpha.c'$.
- *Step on a environment for both $e \to_{\tilde{\mu}} e_1$ and $e \to_{\tilde{\mu}} e_2$, i.e. $e \coloneqq v \cdot e' \to_{\tilde{\mu}} v \cdot e_1' =: e_1$* and $e \to_{\tilde{\mu}} v \cdot e_2' =: e_2$ with $e_1' \ {}_{\tilde{\mu}}\!\leftarrow e' \to_{\tilde{\mu}} e_2'$. By *i.h.*, there is an environment $e_0'$ such that $e_1' \to_{\tilde{\mu}} e_0' \ {}_{\tilde{\mu}}\!\leftarrow e_2'$. According to Remark 35, $e_1 \to_{\tilde{\mu}} e_0 \ {}_{\tilde{\mu}}\!\leftarrow e_2$ by taking $e_0 \coloneqq v \cdot e_0'$.

3. We prove that
   (a) (commands) if $c \to_{\tilde{\mu}} c_1$ and $c \to_{\bar{\lambda}} c_2$ then $c_1 \neq c_2$ and there exists $c'$ such that $c_1 \to_{\bar{\lambda}} c'$ and $c_2 \to_{\tilde{\mu}} c'$;
   (b) (environments) if $e \to_{\tilde{\mu}} e_1$ and $e \to_{\bar{\lambda}} e_2$ then $e_1 \neq e_2$ and there exists $c'$ such that $e_1 \to_{\bar{\lambda}} e'$ and $e_2 \to_{\tilde{\mu}} e'$.

   The proof is by mutual induction on $c$ and $e$ (the proof that $c_1 \neq c_2$ and $e_1 \neq e_2$ is left to the reader). Cases:
   - *Step at the root for $c \to_{\tilde{\mu}} c_1$ and Step on a $\tilde{\mu}$-environment for $c \to_{\bar{\lambda}} c_2$, i.e. $c \coloneqq \langle v \mid \tilde{\mu}x.c_0\rangle \to_{\tilde{\mu}} c_0\{x{\leftarrow}v\} =: c_1$* and $c \to_{\bar{\lambda}} \langle v \mid \tilde{\mu}x.c''\rangle =: c_2$ with $c_0 \to_{\bar{\lambda}} c''$. Then, $c_2 \to_{\tilde{\mu}} c''\{x{\leftarrow}v\} =: c'$. By substitution lemma (Lemma 36), $c_1 \to_{\bar{\lambda}} c'$.
   - *Step on a $v$-environment for $c \to_{\tilde{\mu}} c_1$ and Step at the root for $c \to_{\bar{\lambda}} c_2$, i.e. $c \coloneqq \langle \lambda x.c_0 \mid v \cdot e\rangle \to_{\tilde{\mu}} \langle \lambda x.c_0 \mid v \cdot e'\rangle =: c_1$* with $e \to_{\tilde{\mu}} e'$, and $c \mapsto_{\bar{\lambda}} \langle v \mid (\tilde{\mu}x.c_0)@e\rangle =: c_2$. Then, $c_1 \to_{\bar{\lambda}} \langle v \mid (\tilde{\mu}x.c_0)@e'\rangle =: c'$ and, by append lemma (Lemma 37), $c_2 \to_{\tilde{\mu}} c'$.
   - *Step on an environment for both $c \to_{\tilde{\mu}} c_1$ and $c \to_{\bar{\lambda}} c_2$, i.e. $c \coloneqq \langle v \mid e\rangle \to_{\tilde{\mu}} \langle v \mid e'\rangle =: c_1$* and $c \to_{\bar{\lambda}} \langle v \mid e''\rangle =: c_2$, with $e \to_{\tilde{\mu}} e'$ and $e \to_{\bar{\lambda}} e''$. By *i.h.*, there exists an environment $e_0$ such that $e' \to_{\bar{\lambda}} e_0 \ {}_{\tilde{\mu}}\!\leftarrow e''$, and hence $c_1 \to_{\bar{\lambda}} c' \ {}_{\tilde{\mu}}\!\leftarrow c_2$ by taking $c' \coloneqq \langle v \mid e_0\rangle$, according to Remark 35.
   - *Step on a $\tilde{\mu}$-environment for both $e \to_{\tilde{\mu}} e_1$ and $e \to_{\bar{\lambda}} e_2$, i.e. $e \coloneqq \tilde{\mu}x.c \to_{\tilde{\mu}} \tilde{\mu}x.c_1 =: e_1$* and $e \to_{\bar{\lambda}} \tilde{\mu}x.c_2 =: e_2$, with $c \to_{\tilde{\mu}} c_1$ and $c \to_{\bar{\lambda}} c_2$. By *i.h.*, there exists a command $c_0$ such that $c_1 \to_{\bar{\lambda}} c_0 \ {}_{\tilde{\mu}}\!\leftarrow c_2$, and hence $e_1 \to_{\bar{\lambda}} e' \ {}_{\tilde{\mu}}\!\leftarrow e_2$ by taking $e' \coloneqq \langle v \mid c_0\rangle$, according to Definition 34.
   - *Step on a $v$-environment for both $e \to_{\tilde{\mu}} e_1$ and $e \to_{\bar{\lambda}} e_2$, i.e. $e \coloneqq v \cdot e_0 \to_{\tilde{\mu}} v \cdot e_{01} =: e_1$* and $e \to_{\bar{\lambda}} v \cdot e_{02} =: e_2$, with $e_0 \to_{\tilde{\mu}} e_{01}$ and $e_0 \to_{\bar{\lambda}} e_{02}$. By *i.h.*, there exists an environment $e_0'$ such that $e_{01} \to_{\bar{\lambda}} e_0' \ {}_{\tilde{\mu}}\!\leftarrow e_{02}$, and hence $e_1 \to_{\bar{\lambda}} e' \ {}_{\tilde{\mu}}\!\leftarrow e_2$ by taking $e' \coloneqq v \cdot e_0'$, according to Definition 34.

4. It follows immediately from strong confluence of $\to_{\bar{\lambda}}$ and $\to_{\tilde{\mu}}$ (Propositions 7.1-1), strong commutation of $\to_{\bar{\lambda}}$ and $\to_{\tilde{\mu}}$ (Prop. 7.2) and Hindley-Rosen (Lemma 26). □

### B.2 Proofs of Section 3 (Quantitative Termination Equivalences)

*Simulating $\lambda_{\mathtt{fire}}$ in $\lambda_{\mathtt{vsub}}$*

*Remark 39.* Let $t, u \in \Lambda_{\mathtt{vsub}}$.

1. If $t \equiv u$ then $t{\downarrow} = u{\downarrow}$.

2. If $t \equiv u$ then $t \not\to_{\mathtt{vsub}} u$ (in particular, $t \not\to_{\mathsf{m}} u$ and $t \not\to_{\mathsf{e}} u$).

See p. 13

**Lemma 8** (Simulation of a $\to_{\beta_f}$-Step by $\to_{\mathtt{vsub}}$). *Let $t, u \in \Lambda$.*

1. *If $t \to_{\beta_\lambda} u$ then $t \to_{\mathsf{m}}\to_{\mathsf{e}} u$.*
2. *If $t \to_{\beta_i} u$ then $t \to_{\mathsf{m}}\equiv s$, with $s \in \Lambda_{\mathtt{vsub}}$ clean and $s{\downarrow} = u$.*

*Proof.* Both proofs are by induction on the rewriting step.

1. According to the definition of $t \to_{\beta_\lambda} u$, there are three cases:
   - *Step at the root, i.e.* $t = (\lambda x.s)(\lambda y.r) \mapsto_{\beta_\lambda} s\{x{\leftarrow}\lambda y.r\} = u$: so, $t \to_{\mathsf{m}} s[x{\leftarrow}\lambda y.r] \to_{\mathsf{e}} u$.
   - *Application Left, i.e.* $t = sr \to_{\beta_\lambda} s'r = u$ with $s \to_{\beta_\lambda} s'$: by *i.h.*, $s \to_{\mathsf{m}}\to_{\mathsf{e}} s'$ and hence $t = sr \to_{\mathsf{m}}\to_{\mathsf{e}} s'r = u$.
   - *Application Right, i.e.* $t = sr \to_{\beta_\lambda} sr' = u$ with $r \to_{\beta_\lambda} r'$: by *i.h.*, $r \to_{\mathsf{m}}\to_{\mathsf{e}} r'$ and hence $t = sr \to_{\mathsf{m}}\to_{\mathsf{e}} sr' = u$.
2. According to the definition of $t \to_{\beta_i} u$, there are three cases:
   - *Step at the root, i.e.* $t = (\lambda x.s)i \mapsto_{\beta_i} s\{x{\leftarrow}i\} = u$: then, $t \to_{\mathsf{m}} s[x{\leftarrow}i]$ where $s[x{\leftarrow}i]$ is clean (since $s \in \Lambda$) and $s[x{\leftarrow}i]{\downarrow} = s{\downarrow}\{x{\leftarrow}i{\downarrow}\} = u$ ($s{\downarrow} = s$ and $i{\downarrow} = i$ because $s, i \in \Lambda$). We conclude since $\equiv$ is reflexive.
   - *Application Left, i.e.* $t = sr \to_{\beta_i} s'r = u$ with $s \to_{\beta_i} s'$: by *i.h.*, $s \to_{\mathsf{m}}\equiv q$ where $q$ is a clean $\mathtt{vsub}$-term such that $q{\downarrow} = s'$. So, $q = q_0[x_1{\leftarrow}i_1]\ldots[x_n{\leftarrow}i_n]$ where $q_0 \in \Lambda$ and $i_1, \ldots, i_n$ are inert terms (for some $n \in \mathbb{N}$), moreover we can suppose without loss of generality that $\{x_1, \ldots, x_n\} \cap \mathtt{fv}(r) = \emptyset$. Let $u' = (q_0 r)[x_1{\leftarrow}i_1]\ldots[x_n{\leftarrow}i_n]$: then, $u'$ is a clean $\mathtt{vsub}$-term such that $qr \equiv u'$ and, according to Remark 39.1, $u'{\downarrow} = (qr){\downarrow} = q{\downarrow}r{\downarrow} = s'r = u$. Hence, $t = sr \to_{\mathsf{m}}\equiv qr \equiv u'$ and we conclude since $\equiv$ is transitive.
   - *Application Right, i.e.* $t = sr \to_{\beta_i} sr' = u$ with $r \to_{\beta_i} r'$. Identical to the *application left* case, just switch left and right. □

**Lemma 40 (Fireballs are Closed Under Anti-Substitution of Inert Terms).** *Let $t$ be a $\mathtt{vsub}$-term and $i$ be an inert term.*

1. *If $t\{x{\leftarrow}i\}$ is an abstraction then $t$ is an abstraction.*
2. *If $t\{x{\leftarrow}i\}$ is an inert term then $t$ is an inert term;*
3. *If $t\{x{\leftarrow}i\}$ is a fireball then $t$ is a fireball.*

*Proof.*

1. If $t\{x{\leftarrow}i\} = \lambda y.s$ then there is $r$ such that $s = r\{x{\leftarrow}i\}$, that is $t\{x{\leftarrow}i\} = \lambda y.(r\{x{\leftarrow}i\}) = (\lambda y.r)\{x{\leftarrow}i\}$ and so $t = \lambda y.r$ is an abstraction;
2. By induction on the inert structure of $t\{x{\leftarrow}i\}$. Cases:
   - *Variable, i.e.* $t\{x{\leftarrow}i\} = y$, possibly with $x = y$. Then $t = x$ or $t = y$, and in both cases $t$ is inert.
   - *Compound Inert, i.e.* $t\{x{\leftarrow}i\} = i'f$. If $t$ is a variable then it is inert. Otherwise it is an application $t = us$, and so $u\{x{\leftarrow}i\} = i'$ and $s\{x{\leftarrow}i\} = f$. By *i.h.*, $u$ is an inert term. Consider $f$. Two cases:
     (a) $f$ is an abstraction. Then by Point 1 $s$ is an abstraction.

(b) $f$ is an inert term. Then by *i.h.* $s$ is an inert term.
In both cases $s$ is a fireball, and so $t = us$ is an inert term.

3. Immediate consequence of Lemmas 40.1-2, since every fireball is either an abstraction or an inert term. □

**Lemma 41 (Substitution of Inert Terms Does Not Create $\beta_f$-Redexes).** *Let $t, u$ be terms and $i$ be an inert term. There is $s \in \Lambda$ such that:*

1. *if $t\{x{\leftarrow}i\} \to_{\beta_\lambda} u$ then $t \to_{\beta_\lambda} s$ and $s\{x{\leftarrow}i\} = u$;*
2. *if $t\{x{\leftarrow}i\} \to_{\beta_i} u$ then $t \to_{\beta_i} s$ and $s\{x{\leftarrow}i\} = u$.*

*Proof.* We prove the two points by induction on the evaluation context closing the root redex. Cases:

- *Step at the root*:
  1. *Abstraction Step, i.e.* $t\{x{\leftarrow}i\} := (\lambda y.r\{x{\leftarrow}i\})q\{x{\leftarrow}i\} \mapsto_{\beta_\lambda} r\{x{\leftarrow}i\}\{y{\leftarrow}q\{x{\leftarrow}i\}\} =: u$. By Lemma 40.1, $q$ is an abstraction, since $q\{x{\leftarrow}i\}$ is an abstraction by hypothesis. Then $t = (\lambda y.r)q \mapsto_{\beta_\lambda} r\{y{\leftarrow}q\}$. Then $s := r\{x{\leftarrow}q\}$ verifies the statement, as $s\{x{\leftarrow}i\} = (r\{y{\leftarrow}q\})\{x{\leftarrow}i\} = r\{x{\leftarrow}i\}\{y{\leftarrow}q\{x{\leftarrow}i\}\} = u$.
  2. *Inert Step*, identical to the abstraction subcase, just replace *abstraction* with *inert term* and the use of Lemma 40.1 with the use of Lemma 40.2.
- *Application Left, i.e.* $t = rq$ and reduction takes place in $r$:
  1. *Abstraction Step, i.e.* $t\{x{\leftarrow}i\} := r\{x{\leftarrow}i\}q\{x{\leftarrow}i\} \to_{\beta_\lambda} pq\{x{\leftarrow}i\} =: u$. By *i.h.* there exists $s' \in \Lambda$ such that $p = s'\{x{\leftarrow}i\}$ and $r \to_{\beta_\lambda} s'$. Then $s := s'q$ satisfies the statement, as $s\{x{\leftarrow}i\} = (s'q)\{x{\leftarrow}i\} = s'\{x{\leftarrow}i\}q\{x{\leftarrow}i\} = u$.
  2. *Inert Step*, identical to the abstraction subcase.
- *Application Right, i.e.* $t = rq$ and reduction takes place in $q$. Identical to the *application left* case, just switch left and right. □

See p. 13

**Lemma 9** (Projection of a $\beta_f$-Step on $\to_{\mathtt{vsub}}$ via Unfolding)**.** *Let $t$ be a clean* vsub*-term and $u$ be a term.*

1. *If $t{\downarrow} \to_{\beta_\lambda} u$ then $t \to_{\mathsf{m}}\to_{\mathsf{e}} s$, with $s \in \Lambda_{\mathtt{vsub}}$ clean s.t. $s{\downarrow} = u$.*
2. *If $t{\downarrow} \to_{\beta_i} u$ then $t \to_{\mathsf{m}}\equiv s$, with $s \in \Lambda_{\mathtt{vsub}}$ clean s.t. $s{\downarrow} = u$.*

*Proof.* Since $t$ is clean, there are a $\lambda$-term $q$ and some inert $\lambda$-terms $i_1, \dots, i_n$ (with $n \in \mathbb{N}$) such that $t = q[x_1{\leftarrow}i_1]\dots[x_n{\leftarrow}i_n]$. We prove both points by induction on $n \in \mathbb{N}$. The base case (*i.e.* $n = 0$) is given by the simulation of one-step reductions given by Lemma 8, since $t = q \in \Lambda$ and hence $t{\downarrow} = t$ (recall that, when applying Lemma 8.1, $u \in \Lambda$ implies that $u$ is clean and $u{\downarrow} = u$).

Consider now $n > 0$. Let $t_{n-1} := q[x_1{\leftarrow}i_1]\dots[x_{n-1}{\leftarrow}i_{n-1}]$: so, $t = t_{n-1}[x_n{\leftarrow}i_n]$ and $t{\downarrow} = t_{n-1}{\downarrow}\{x_n{\leftarrow}i_n\}$. Both points rely on the fact that the substitution of inert terms cannot create redexes (Lemma 41). Namely,

1. $\beta_\lambda$*-step*: the application of Lemma 41.1 to $t{\downarrow} = t_{n-1}{\downarrow}\{x_n{\leftarrow}i_n\} \to_{\beta_\lambda} u$ (since $t_{n-1}{\downarrow} \in \Lambda$, *i.e.* it has no ES) provides $r \in \Lambda$ such that $t_{n-1}{\downarrow} \to_{\beta_\lambda} r$ and $r\{x_n{\leftarrow}i_n\} = u$. By *i.h.*, $t_n \to_{\mathsf{m}}\to_{\mathsf{e}} s$ where $s$ is a clean vsub-term such that $s{\downarrow} = r$, and thus $t = t_{n-1}[x_n{\leftarrow}i_n] \to_{\mathsf{m}}\to_{\mathsf{e}} s[x_n{\leftarrow}i_n]$. Moreover, $s[x_n{\leftarrow}i_n]$ is clean and $s[x_n{\leftarrow}i_n]{\downarrow} = s{\downarrow}\{x_n{\leftarrow}i_n\} = r\{x_n{\leftarrow}i_n\} = u$.

2. $\beta_i$-*step*: the application of Lemma 41.2 to $t\!\downarrow = t_{n-1}\!\downarrow\{x_n\leftarrow i_n\} \to_{\beta_i} u$ provides $r \in \Lambda$ such that $t_{n-1}\!\downarrow \to_{\beta_i} r$ and $r\{x_n\leftarrow i_n\} = u$. By *i.h.*, $t_{n-1} \to_{\mathsf{m}}\equiv s$ where $s$ is a clean $\mathtt{vsub}$-term such that $s\!\downarrow = r$; thus, $t = t_{n-1}[x_n\leftarrow i_n] \to_{\mathsf{m}}\equiv s[x_n\leftarrow i_n]$. Moreover, $s[x_n\leftarrow i_n]$ is clean and $s[x_n\leftarrow i_n]\!\downarrow = s\!\downarrow\{x_n\leftarrow i_n\} = r\{x_n\leftarrow i_n\} = u$. □

**Lemma 10.** *Let $t$ be a clean $\mathtt{vsub}$-term. If $t\!\downarrow$ is a fireball, then $t$ is $\{\mathtt{m}, \mathsf{e}_\lambda\}$-normal and its body is a fireball.* See p. 14

*Proof.* First, we prove that if $t\!\downarrow$ is a fireball then for some fireball $f$ and inert terms $i_1, \dots i_n$ one has $t = f[x_1\leftarrow i_1]\dots[x_n\leftarrow i_n]$. Since $t$ is clean, there are a $\lambda$-term $u$ and some inerts $\lambda$-terms $i_1, \dots, i_n$ (with $n \in \mathbb{N}$) such that $t = u[x_1\leftarrow i_1]\dots[x_n\leftarrow i_n]$. We prove by induction on $n \in \mathbb{N}$ that $u$ is a fireball.

If $n = 0$, then $t = u \in \Lambda$, thus $u = t\!\downarrow$ and hence $u$ is a fireball.

Suppose $n > 0$ and let $s \coloneqq u[x_1\leftarrow i_1]\dots[x_{n-1}\leftarrow i_{n-1}]$, which is a clean $\mathtt{vsub}$-term: then, $t = s[x_n\leftarrow i_n]$ and hence $t\!\downarrow = s\!\downarrow\{x_n\leftarrow i_n\}$ (as $i_n\!\downarrow = i_n$ because $i_n \in \Lambda$). By Lemma 40.3, $s\!\downarrow$ is a fireball. By *i.h.*, $u$ is a fireball.

Now, fireballs are $\mathtt{vsub}$-normal. Indeed, a fireball is without ES, hence it is without $\mathsf{e}$-redexes, moreover it is immediate to prove that fireballs are $\mathsf{m}$-normal (by simply adapting the proof of Lemma 28).

So, $t = f[x_1\leftarrow i_1]\dots[x_n\leftarrow i_n]$ can only have $\mathsf{e}_y$-redexes. □

**Lemma 11** (Linear Postponement of $\to_{\mathsf{e}_y}$)**.** *Let $t, u, s \in \Lambda_{\mathtt{vsub}}$.* See p. 14

1. *If $t \to_{\mathsf{e}_y} s \to_{\mathsf{m}} u$ then $t \to_{\mathsf{m}}\to_{\mathsf{e}_y} u$.*
2. *If $t \to_{\mathsf{e}_y}\to_{\mathsf{e}_\lambda} u$ then $t \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}} u$.*
3. *If $d\colon t \to^*_{\mathtt{vsub}} u$ then $e\colon t \to^*_{\mathtt{m},\mathsf{e}_\lambda}\to^*_{\mathsf{e}_y} u$ with $|e|_{\mathtt{vsub}} = |d|_{\mathtt{vsub}}$, $|e|_{\mathsf{m}} = |d|_{\mathsf{m}}$, $|e|_{\mathsf{e}} = |d|_{\mathsf{e}}$, and $|e|_{\mathsf{e}_\lambda} \geq |d|_{\mathsf{e}_\lambda}$.*

*Proof.* 1. By induction on the definition of $t \to_{\mathsf{e}_y} s$. Since the $\mathsf{e}_y$-step cannot create in $s$ new $\mathsf{m}$-redexes not occurring in $t$, the $\mathsf{m}$-redex fired in $s \to_{\mathsf{m}} u$ is (a residual of a $\mathsf{m}$-redex) already occurring in $t$. So, there are the following cases.

- *Step at the Root for $t \to_{\mathsf{e}_y} s$ and ES Left for $s \to_{\mathsf{m}} u$, i.e.* $t \coloneqq r[z\leftarrow L\langle x\rangle] \to_{\mathsf{e}_y} L\langle r\{z\leftarrow x\}\rangle =: s$ and $s \to_{\mathsf{m}} L\langle r'\{z\leftarrow x\}\rangle =: u$ with $r \to_{\mathsf{m}} r'$: then $t \to_{\mathsf{m}} r'[z\leftarrow L\langle x\rangle] \to_{\mathsf{e}_y} u$;
- *Step at the Root for $t \to_{\mathsf{e}_y} s$ and ES "quasi-Right" for $s \to_{\mathsf{m}} u$, i.e.* $t \coloneqq r[z\leftarrow x[x_1\leftarrow t_1]\dots[x_n\leftarrow t_n]] \to_{\mathsf{e}_y} r\{z\leftarrow x\}[x_1\leftarrow t_1]\dots[x_n\leftarrow t_n] =: s$ and
$$t \to_{\mathsf{m}} r[z\leftarrow x[x_1\leftarrow t_1]\dots[x_j\leftarrow t'_j]\dots[x_n\leftarrow t_n]] =: u$$
for some $n > 0$, and $t_j \to_{\mathsf{m}} t'_j$ for some $1 \leq j \leq n$: then, $t \to_{\mathsf{m}} r[z\leftarrow x[x_1\leftarrow t_1]\dots[x_j\leftarrow t'_j]\dots[x_n\leftarrow t_n]] \to_{\mathsf{e}_y} u$;
- *Application Left for $t \to_{\mathsf{e}_y} s$* and *Application Right for $s \to_{\mathsf{m}} u$, i.e.* $t \coloneqq rq \to_{\mathsf{e}_y} r'q =: s$ and $s \to_{\mathsf{m}} r'q' =: u$ with $r \to_{\mathsf{e}_y} r'$ and $q \to_{\mathsf{m}} q'$: then, $t \to_{\mathsf{m}} rq' \to_{\mathsf{e}_y} u$;
- *Application Left for both $t \to_{\mathsf{e}_y} s$ and $s \to_{\mathsf{m}} u$, i.e.* $t \coloneqq rq \to_{\mathsf{e}_y} r'q =: s$ and $s \to_{\mathsf{m}} r''q =: u$ with $r \to_{\mathsf{e}_y} r'$ and $r' \to_{\mathsf{m}} r''$: by *i.h.*, $r \to_{\mathsf{m}}\to_{\mathsf{e}_y} r''$, hence $t \to_{\mathsf{m}}\to_{\mathsf{e}_y} u$;

– *Application Left for* $t \to_{\mathsf{e}_y} s$ and *Step at the Root for* $s \to_{\mathsf{m}} u$, *i.e.* $t := (\lambda x.q)[x_1{\leftarrow}t_1]\ldots[x_n{\leftarrow}t_n]r \to_{\mathsf{e}_y} (\lambda x.q)[x_1{\leftarrow}t_1]\ldots[x_j{\leftarrow}t_j']\ldots[x_n{\leftarrow}t_n]r =: s$ with $n > 0$ and $t_j \to_{\mathsf{e}_y} t_j'$ for some $1 \leq j \leq n$, and

$$s \to_{\mathsf{m}} q[x{\leftarrow}r][x_1{\leftarrow}t_1]\ldots[x_j{\leftarrow}t_j']\ldots[x_n{\leftarrow}t_n] =: u$$

then,

$$t \to_{\mathsf{m}} q[x{\leftarrow}r][x_1{\leftarrow}t_1]\ldots[x_n{\leftarrow}t_n] \to_{\mathsf{e}_y} u;$$

– *Application Right for* $t \to_{\mathsf{e}_y} s$ and *Application Left for* $s \to_{\mathsf{m}} u$, *i.e.* $t := qr \to_{\mathsf{e}_y} qr' =: s$ and $s \to_{\mathsf{m}} q'r' =: u$ with $r \to_{\mathsf{e}_y} r'$ and $q \to_{\mathsf{m}} q'$: then, $t \to_{\mathsf{m}} q'r \to_{\mathsf{e}_y} u$;
– *Application Right for both* $t \to_{\mathsf{e}_y} s$ *and* $s \to_{\mathsf{m}} u$, *i.e.* $t := qr \to_{\mathsf{e}_y} qr' =: s$ and $s \to_{\mathsf{m}} qr'' =: u$ with $r \to_{\mathsf{e}_y} r'$ and $r' \to_{\mathsf{m}} r''$: by *i.h.*, $r \to_{\mathsf{m}}\to_{\mathsf{e}_y} r''$, hence $t \to_{\mathsf{m}}\to_{\mathsf{e}_y} u$;
– *Application Right for* $t \to_{\mathsf{e}_y} s$ and *Step at the Root for* $s \to_{\mathsf{m}} u$, *i.e.* $t := L\lambda x.qr \to_{\mathsf{e}_y} L\langle\lambda x.q\rangle r' =: s$ with $r \to_{\mathsf{e}_y} r'$, and $s \to_{\mathsf{m}} L\langle q[x{\leftarrow}r']\rangle =: u$: then, $t \to_{\mathsf{m}} L\langle q[x{\leftarrow}r]\rangle \to_{\mathsf{e}_y} u$;
– *ES Left for* $t \to_{\mathsf{e}_y} s$ and *ES Right for* $s \to_{\mathsf{m}} u$, *i.e.* $t := r[x{\leftarrow}q] \to_{\mathsf{e}_y} r'[x{\leftarrow}q] =: s$ and $s \to_{\mathsf{m}} r'[x{\leftarrow}q'] =: u$ with $r \to_{\mathsf{e}_y} r'$ and $q \to_{\mathsf{m}} q'$: then, $t \to_{\mathsf{m}} r[x{\leftarrow}q'] \to_{\mathsf{e}_y} u$;
– *ES Left for both* $t \to_{\mathsf{e}_y} s$ *and* $s \to_{\mathsf{m}} u$, *i.e.* $t := r[x{\leftarrow}q] \to_{\mathsf{e}_y} r'[x{\leftarrow}q] =: s$ and $s \to_{\mathsf{m}} r''[x{\leftarrow}q] =: u$ with $r \to_{\mathsf{e}_y} r'$ and $r' \to_{\mathsf{m}} r''$: by *i.h.*, $r \to_{\mathsf{m}}\to_{\mathsf{e}_y} r''$, hence $t \to_{\mathsf{m}}\to_{\mathsf{e}_y} u$;
– *ES Right for* $t \to_{\mathsf{e}_y} s$ and *ES Left for* $s \to_{\mathsf{m}} u$, *i.e.* $t := q[x{\leftarrow}r] \to_{\mathsf{e}_y} q[x{\leftarrow}r'] =: s$ and $s \to_{\mathsf{m}} q'[x{\leftarrow}r'] =: u$ with $r \to_{\mathsf{e}_y} r'$ and $q \to_{\mathsf{m}} q'$: then, $t \to_{\mathsf{m}} q'[x{\leftarrow}r] \to_{\mathsf{e}_y} u$;
– *ES Right for both* $t \to_{\mathsf{e}_y} s$ *and* $s \to_{\mathsf{m}} u$, *i.e.* $t := q[x{\leftarrow}r] \to_{\mathsf{e}_y} q[x{\leftarrow}r'] =: s$ and $s \to_{\mathsf{m}} q[x{\leftarrow}r''] =: u$ with $r \to_{\mathsf{e}_y} r'$ and $r' \to_{\mathsf{m}} r''$: by *i.h.*, $r \to_{\mathsf{m}}\to_{\mathsf{e}_y} r''$, hence $t \to_{\mathsf{m}}\to_{\mathsf{e}_y} u$.

2. By induction on the definition of $t \to_{\mathsf{e}_y} s$. Since the $\mathsf{e}_y$-step cannot create in $s$ new $\mathsf{e}_\lambda$-redexes not occurring in $t$, the $\mathsf{e}_\lambda$-redex fired in $s \to_{\mathsf{e}_\lambda} u$ is (a residual of a $\mathsf{e}_\lambda$-redex) already occurring in $t$. So, there are the following cases.
   – *Step at the Root for both* $t \to_{\mathsf{e}_y} s$ *and* $s \to_{\mathsf{e}_\lambda} u$, *i.e.*

$$t := r[x{\leftarrow}L'\langle z\rangle[y{\leftarrow}L\langle\lambda x.q\rangle]] \to_{\mathsf{e}_y} L'\langle r\{x{\leftarrow}z\}\rangle[y{\leftarrow}L\langle\lambda x.q\rangle] =: s$$

   and $s \to_{\mathsf{e}_\lambda} L\langle L'\langle r\{x{\leftarrow}z\}\rangle\{y{\leftarrow}\lambda x.q\}\rangle =: u$ (with possibly $y = z$). We set $L'' := L'\{y{\leftarrow}\lambda x.q\}$ *i.e.* $L''$ is the substitution context obtained from $L'$ by the capture-avoiding substitution of $\lambda x.q$ for each free occurrence of $y$ in $L'$. We can suppose without loss of generality that $y \notin \mathtt{fv}(L) \cup \mathtt{fv}(r)$. There are two sub-cases:
   • either $y = z$ and then $t \to_{\mathsf{e}_\lambda} r[x{\leftarrow}L\langle L''\langle\lambda x.q\rangle\rangle] \to_{\mathsf{e}_\lambda} L\langle L''\langle r\{x{\leftarrow}\lambda x.q\}\rangle\rangle = u$,
   • or $y \neq z$ and then $t \to_{\mathsf{e}_\lambda} r[x{\leftarrow}L\langle L''\langle z\rangle\rangle] \to_{\mathsf{e}_y} L\langle L''\langle r\{x{\leftarrow}z\}\rangle\rangle = u$.
   – *Step at the Root for* $t \to_{\mathsf{e}_y} s$ *and ES Left for* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t := r[z{\leftarrow}L\langle x\rangle] \to_{\mathsf{e}_y} L\langle r\{z{\leftarrow}x\}\rangle =: s$ and $s \to_{\mathsf{e}_\lambda} L\langle r'\{z{\leftarrow}x\}\rangle =: u$ with $r \to_{\mathsf{e}_\lambda} r'$: then $t \to_{\mathsf{e}_\lambda} r'[z{\leftarrow}L\langle x\rangle] \to_{\mathsf{e}_y} u$;

- *Step at the Root for* $t \to_{\mathsf{e}_y} s$ *and ES "quasi-Right" for* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq r[z{\leftarrow}x[x_1{\leftarrow}t_1]\ldots[x_n{\leftarrow}t_n]] \to_{\mathsf{e}_y} r\{z{\leftarrow}x\}[x_1{\leftarrow}t_1]\ldots[x_n{\leftarrow}t_n] \eqqcolon s$ for some $n > 0$, and $t_j \to_{\mathsf{e}_\lambda} t'_j$ for some $1 \leq j \leq n$, and
$$s \to_{\mathsf{e}_\lambda} r\{z{\leftarrow}x\}[x_1{\leftarrow}t_1]\ldots[x_j{\leftarrow}t'_j]\ldots[x_n{\leftarrow}t_n] \eqqcolon u$$
then, $t \to_{\mathsf{e}_\lambda} r[z{\leftarrow}x[x_1{\leftarrow}t_1]\ldots[x_j{\leftarrow}t'_j]\ldots[x_n{\leftarrow}t_n]] \to_{\mathsf{e}_y} u$;
- *Application Left for* $t \to_{\mathsf{e}_y} s$ *and Application Right for* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq rq \to_{\mathsf{e}_y} r'q \eqqcolon s$ and $s \to_{\mathsf{e}_\lambda} r'q' \eqqcolon u$ with $r \to_{\mathsf{e}_y} r'$ and $q \to_{\mathsf{e}_\lambda} q'$: then, $t \to_{\mathsf{e}_\lambda} rq' \to_{\mathsf{e}_y} u$;
- *Application Left for both* $t \to_{\mathsf{e}_y} s$ *and* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq rq \to_{\mathsf{e}_y} r'q \eqqcolon s$ and $s \to_{\mathsf{e}_\lambda} r''q \eqqcolon u$ with $r \to_{\mathsf{e}_y} r'$ and $r' \to_{\mathsf{e}_\lambda} r''$: by *i.h.*, $r \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}} r''$, hence $t \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}} u$;
- *Application Right for* $t \to_{\mathsf{e}_y} s$ *and Application Left for* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq qr \to_{\mathsf{e}_y} qr' \eqqcolon s$ and $s \to_{\mathsf{e}_\lambda} q'r' \eqqcolon u$ with $r \to_{\mathsf{e}_y} r'$ and $q \to_{\mathsf{e}_\lambda} q'$: then, $t \to_{\mathsf{e}_\lambda} q'r \to_{\mathsf{e}_y} u$;
- *Application Right for both* $t \to_{\mathsf{e}_y} s$ *and* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq qr \to_{\mathsf{e}_y} qr' \eqqcolon s$ and $s \to_{\mathsf{e}_\lambda} qr'' \eqqcolon u$ with $r \to_{\mathsf{e}_y} r'$ and $r' \to_{\mathsf{e}_\lambda} r''$: by *i.h.*, $r \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}} r''$, hence $t \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}} u$;
- *ES Left for* $t \to_{\mathsf{e}_y} s$ *and Step at the Root for* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq r[z{\leftarrow}L\langle\lambda y.q\rangle] \to_{\mathsf{e}_y} r'[z{\leftarrow}L\langle\lambda y.q\rangle] \eqqcolon s$ and $s \to_{\mathsf{e}_\lambda} L\langle r'\{z{\leftarrow}\lambda y.q\}\rangle \eqqcolon u$ with $r \to_{\mathsf{e}_y} r'$: this means that in $r$ there is an ES of the form $[y{\leftarrow}x]$ (possibly $x = z$) which is fired in $r \to_{\mathsf{e}_y} r'$; then, $t \to_{\mathsf{e}_\lambda} L\langle r\{z{\leftarrow}\lambda y.q\}\rangle \to_{\mathsf{e}} u$, where the last $\mathsf{e}$-step is a $\mathsf{e}_\lambda$-step if $x = z$, otherwise it is a $\mathsf{e}_y$-step;
- *ES Left for* $t \to_{\mathsf{e}_y} s$ *and ES Right for* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq r[x{\leftarrow}q] \to_{\mathsf{e}_y} r'[x{\leftarrow}q] \eqqcolon s$ and $s \to_{\mathsf{e}_\lambda} r'[x{\leftarrow}q'] \eqqcolon u$ with $r \to_{\mathsf{e}_y} r'$ and $q \to_{\mathsf{e}_\lambda} q'$: then, $t \to_{\mathsf{e}_\lambda} r[x{\leftarrow}q'] \to_{\mathsf{e}_y} u$;
- *ES Left for both* $t \to_{\mathsf{e}_y} s$ *and* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq r[x{\leftarrow}q] \to_{\mathsf{e}_y} r'[x{\leftarrow}q] \eqqcolon s$ and $s \to_{\mathsf{e}_\lambda} r''[x{\leftarrow}q] \eqqcolon u$ with $r \to_{\mathsf{e}_y} r'$ and $r' \to_{\mathsf{e}_\lambda} r''$: by *i.h.*, $r \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}} r''$, so $t \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}} u$;
- *ES Right for* $t \to_{\mathsf{e}_y} s$ *and Step at the Root for* $s \to_{\mathsf{e}_\lambda} u$, *i.e.*
$$\begin{aligned} t &\coloneqq r[z{\leftarrow}(\lambda y.q)[x_1{\leftarrow}t_1]\ldots[x_n{\leftarrow}t_n]] \\ &\to_{\mathsf{e}_y} r[z{\leftarrow}(\lambda y.q)[x_1{\leftarrow}t_1]\ldots[x_j{\leftarrow}t'_j]\ldots[x_n{\leftarrow}t_n]] \eqqcolon s \end{aligned}$$
for some $n > 0$, and $t_j \to_{\mathsf{e}_y} t'_j$ for some $1 \leq j \leq n$, and $s \to_{\mathsf{e}_\lambda} r\{z{\leftarrow}\lambda y.q\}[x_1{\leftarrow}t_1]\ldots[x_j{\leftarrow}t'_j]\ldots[x_n{\leftarrow}t_n] \eqqcolon u$: then,
$$t \to_{\mathsf{e}_\lambda} r\{z{\leftarrow}\lambda y.q\}[x_1{\leftarrow}t_1]\ldots[x_n{\leftarrow}t_n] \to_{\mathsf{e}_y} u\,;$$
- *ES Right for* $t \to_{\mathsf{e}_y} s$ *and ES Left for* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq q[x{\leftarrow}r] \to_{\mathsf{e}_y} q[x{\leftarrow}r'] \eqqcolon s$ and $s \to_{\mathsf{e}_\lambda} q'[x{\leftarrow}r'] \eqqcolon u$ with $r \to_{\mathsf{e}_y} r'$ and $q \to_{\mathsf{e}_\lambda} q'$: then, $t \to_{\mathsf{e}_\lambda} q'[x{\leftarrow}r] \to_{\mathsf{e}_y} u$;
- *ES Right for both* $t \to_{\mathsf{e}_y} s$ *and* $s \to_{\mathsf{e}_\lambda} u$, *i.e.* $t \coloneqq q[x{\leftarrow}r] \to_{\mathsf{e}_y} q[x{\leftarrow}r'] \eqqcolon s$ and $s \to_{\mathsf{e}_\lambda} q[x{\leftarrow}r''] \eqqcolon u$ with $r \to_{\mathsf{e}_y} r'$ and $r' \to_{\mathsf{e}_\lambda} r''$: by *i.h.*, $r \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}} r''$, hence $t \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}} u$.

3. By induction on $|d|_{\mathtt{vsub}} \in \mathbb{N}$, using Lemmas 11.1-2 in the inductive case. □

**Theorem 12** (Quantitative Simulation of $\lambda_{\mathtt{fire}}$ in $\lambda_{\mathtt{vsub}}$)**.** *Let $t, u \in \Lambda$. If $d\colon t \to^*_{\beta_f} u$ then there are $s, r \in \Lambda_{\mathtt{vsub}}$ and $e\colon t \to^*_{\mathtt{vsub}} r$ such that* See p. 14

1. Qualitative Relationship*: $r \equiv s$, $u = s{\downarrow} = r{\downarrow}$ and $s$ is clean;*
2. Quantitative Relationship*:*
   *(a)* Multiplicative Steps: $|d|_{\beta_f} = |e|_{\mathtt{m}}$*;*
   *(b)* Exponential (Abstraction) Steps: $|d|_{\beta_\lambda} = |e|_{\mathtt{e}_\lambda} = |e|_{\mathtt{e}}$.
3. Normal Forms*: if $u$ is $\beta_f$-normal then there exists $f\colon r \to^*_{\mathtt{e}_y} q$ such that $q$ is a* vsub*-normal form and $|f|_{\mathtt{e}_y} \leq |e|_{\mathtt{m}} - |e|_{\mathtt{e}_\lambda}$.*

*Proof.* The first two points are proved together.

1-2. By the remark at the beginning of this section of the Appendix (Remarks 39.1-2), it is sufficient to show that there exists $e\colon t \to^*_{\mathtt{vsub}} \equiv s \in \Lambda_{\mathtt{vsub}}$ such that $u = s{\downarrow}$ with $s$ clean, and $|d|_{\beta_f} = |e|_{\mathtt{m}}$ and $|d|_{\beta_\lambda} = |e|_{\mathtt{e}_\lambda}$ (the fact that $|e|_{\mathtt{e}_\lambda} = |e|_{\mathtt{e}}$ is immediate, since the simulation obtained by iterating the projection in Lemma 9 never uses $\to_{\mathtt{e}_y}$). We proceed by induction on $|d|_{\beta_f} \in \mathbb{N}$. Cases:
   - *Empty derivation, i.e.* $|d|_{\beta_f} = 0$ then $t = u$ and $|d|_{\beta_\lambda} = 0$, so we conclude taking $s := u$ and $e$ as the empty derivation.
   - *Non-empty derivation, i.e.* $|d|_{\beta_f} > 0$: then, $d\colon t \to^*_{\beta_f} r \to_{\beta_f} u$ and let $d'\colon t \to^*_{\beta_f} r$ be the derivation obtained from $d$ by removing its last step $r \to_{\beta_f} u$. By *i.h.*, there is $e'\colon t \to^*_{\mathtt{vsub}} \equiv q$ such that $r = q{\downarrow}$, $q$ is clean, $|d'|_{\beta_f} = |e'|_{\mathtt{m}}$, and $|d'|_{\beta_\lambda} = |e'|_{\mathtt{e}_\lambda}$. By applying Lemma 9 to the last step $r \to_{\beta_f} u$ of $d$, we obtain $s$ such that either $q \to_{\mathtt{m}} \to_{\mathtt{e}} s$, if $q{\downarrow} \to_{\beta_v} u$, or $q \to_{\mathtt{m}} \equiv s$, if $q{\downarrow} \to_{\beta_i} u$, and in both cases $s$ is a clean vsub-term such that $s'{\downarrow} = u$. Note that both cases can be summed up with $q \to_{\mathtt{m}} \overset{\equiv}{\to}_{\mathtt{e}} s$. Composing the two obtained derivations $e'\colon t \to^*_{\mathtt{vsub}} \equiv q$ and $q \to_{\mathtt{m}} \overset{\equiv}{\to}_{\mathtt{e}} s$, we obtain the derivation $e''\colon t \to^*_{\mathtt{vsub}} \equiv q \to_{\mathtt{m}} \overset{\equiv}{\to}_{\mathtt{e}} s$ that satisfies the quantitative relationships but not yet the qualitative one, as $\equiv$ appears between two steps of $e''$. It is then enough to apply the strong bisimulation property of $\equiv$ (Lemma 5.2), that provides a derivation $e : t \to^*_{\mathtt{vsub}} \to_{\mathtt{m}} \overset{\equiv}{\to}_{\mathtt{e}} \equiv s$ with the same quantitative properties of $e''$.

3. If $u$ is $\beta_f$-normal then it is a fireball (by open harmony, Prop. 2) and so $s$ is $\{\mathtt{m}, \mathtt{e}_\lambda\}$-normal by Lemma 10. By Prop. 4.1, $\to_{\mathtt{e}_y}$ terminates and so there are $p$ and a derivation $g\colon s \to^*_{\mathtt{e}_y} p$ such that $p$ is a $\mathtt{e}_y$-normal form. If $p$ is not a vsub-normal form, then it has a $\{\mathtt{m}, \mathtt{e}_\lambda\}$-redex, but by postponement of $\to_{\mathtt{e}_y}$ (Lemma 11) such a redex was already in $s$, against hypothesis. So $p$ is a vsub-normal form. Then we have $r \equiv s \to^*_{\mathtt{e}_y} p$. Postponing $\equiv$ (Lemmas 5.2-3), we obtain that there exists a vsub-normal form $q$ and a derivation $f\colon r \to^*_{\mathtt{e}_y} q \equiv p$.
   To estimate the length of $f$ consider $e$ followed by $f$, *i.e.* $e; f\colon t \to^*_{\mathtt{m},\mathtt{e}_\lambda} r \to^*_{\mathtt{e}_y} q$. By Prop. 4.4, $|e; f|_{\mathtt{e}} \leq |e; f|_{\mathtt{m}} = |e|_{\mathtt{m}}$, and since $|e; f|_{\mathtt{e}} = |e; f|_{\mathtt{e}_\lambda} + |e; f|_{\mathtt{e}_y} = |e|_{\mathtt{e}_\lambda} + |f|_{\mathtt{e}_y}$ we obtain $|e|_{\mathtt{e}_\lambda} + |f|_{\mathtt{e}_y} \leq |e|_{\mathtt{m}}$, *i.e.* $|f|_{\mathtt{e}_y} \leq |e|_{\mathtt{m}} - |e|_{\mathtt{e}_\lambda}$. □

See p. 14 **Corollary 13** (Linear Termination Equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{fire}}$)**.** *Let $t \in$*

*4. There exists a $\beta_f$-normalizing derivation $d$ from $t$ iff there exists a* vsub*-normalizing derivation $e$ from $t$. Moreover, $|d|_{\beta_f} \leq |e|_{\mathtt{vsub}} \leq 2|d|_{\beta_f}$, i.e. they are linearly related.*

*Proof.*

**⇒:** Let $d\colon t \to^*_{\beta_f} u$ be a $\beta_f$-normalizing derivation and $e\colon t \to^*_{\mathtt{vsub}} \to^*_{\mathtt{e}_y} q$ be the composition of its projection in $\lambda_{\mathtt{vsub}}$ with the extension to a $\mathtt{e}_y$-derivation with $q$ vsub-normal, according to Thm. 12. Then $e$ is a vsub-normalizing derivation from $t$.

**⇐:** By contradiction, suppose that there is a diverging $\beta_f$-derivation from $t$ in $\lambda_{\mathtt{fire}}$. By Thm. 12 it projects to a vsub-derivation in $\lambda_{\mathtt{vsub}}$ that is at least as long as the one in $\lambda_{\mathtt{fire}}$, absurd.

About lengths, $|d|_{\beta_f} \leq |e|_{\mathtt{vsub}}$ since $|e|_{\mathtt{m}} = |d|_{\beta_f}$ (Thm. 12.2). By Prop. 4.4, $|e|_{\mathtt{e}} \leq |e|_{\mathtt{m}}$ and so $|e|_{\mathtt{vsub}} = |e|_{\mathtt{m}} + |e|_{\mathtt{e}} \leq 2|d|_{\beta_f}$. □

*Simulating $\lambda_{\mathtt{sh}}$ in $\lambda_{\mathtt{vsub}}$*

**Lemma 42 (Simulation of a sh-Step on $\lambda_{\mathtt{vsub}}$).** *Let $t, u \in \Lambda$.*

1. *If $t \to_{\sigma^\flat} u$ then there exist $s, r \in \Lambda_{\mathtt{vsub}}$ s.t. $t \to^+_{\mathtt{m}} s \equiv r \ {}_{\mathtt{m}}^{+}\!\!\leftarrow u$.*
2. *If $t \to_{\beta_v^\flat} u$ then there exists $s \in \Lambda_{\mathtt{vsub}}$ s.t. $t \to^+_{\mathtt{m}} \to_{\mathtt{e}} s \ {}_{\mathtt{m}}^{*}\!\!\leftarrow u$.*

*Proof.* 1. By induction on the definition of $t \to_{\sigma^\flat} s$, following Remark 32. There are four cases:

(a) *Step at the root, i.e.* $t \mapsto_\sigma u$.
   i. either $t \coloneqq (\lambda x.q)sr \mapsto_{\sigma_1} (\lambda x.qr)s \eqqcolon u$ with $x \notin \mathtt{fv}(r)$, and then $t = (\lambda x.q)sr \to_{\mathtt{m}} q[x{\leftarrow}s]r \equiv (qr)[x{\leftarrow}s] \ {}_{\mathtt{m}}\!\!\leftarrow (\lambda x.qr)s = u$;
   ii. or $t \coloneqq v((\lambda x.s)r) \mapsto_{\sigma_3} (\lambda x.vs)r \eqqcolon u$ with $x \notin \mathtt{fv}(v)$ and then $t = v((\lambda x.s)r) \to_{\mathtt{m}} v(s[x{\leftarrow}r]) \equiv (vs)[x{\leftarrow}r] \ {}_{\mathtt{m}}\!\!\leftarrow (\lambda x.vs)r = u$.

(b) *Application Left, i.e.* $t \coloneqq sr \to_{\sigma^\flat} qr \eqqcolon u$ with $s \to_{\sigma^\flat} q$. The result follows by the *i.h.*, as $\to_{\mathtt{m}}$ and $\equiv$ are closed by applicative contexts.

(c) *Application Right, i.e.* $t \coloneqq sr \to_{\sigma^\flat} sq \eqqcolon u$ with $r \to_{\sigma^\flat} q$. The result follows by the *i.h.*, as $\to_{\mathtt{m}}$ and $\equiv$ are closed by applicative contexts.

(d) *Inside a $\beta$-context, i.e.* $t \coloneqq (\lambda x.s)r \to_{\sigma^\flat} (\lambda x.q)r \eqqcolon u$ with $s \to_{\sigma^\flat} q$. By *i.h.*, $s \to^+_{\mathtt{m}} s' \equiv q' \ {}_{\mathtt{m}}^{+}\!\!\leftarrow q$. Now, $\to_{\mathtt{m}}$ and $\equiv$ are not closed by balanced contexts, but it is enough to apply a further $\to_{\mathtt{m}}$ step to the balanced context (as $\to_{\mathtt{m}}$ and $\equiv$ are instead closed by substitution contexts), obtaining $t = (\lambda x.s)r \to_{\mathtt{m}} s[x{\leftarrow}r] \to^+_{\mathtt{m}} s'[x{\leftarrow}r] \equiv q'[x{\leftarrow}r] \ {}_{\mathtt{m}}^{+}\!\!\leftarrow q[x{\leftarrow}r] \ {}_{\mathtt{m}}\!\!\leftarrow (\lambda x.q)r = u$.

2. By induction on the definition of $t \to_{\beta_v^\flat} u$, there are four cases:

(a) *Step at the root, i.e.* $t = (\lambda x.r)v \mapsto_{\beta_v} r\{x{\leftarrow}v\} = u$. So, $t \to_{\mathtt{m}} r[x{\leftarrow}v] \to_{\mathtt{e}} u$.

(b) *Application Left.* It follows by the *i.h.*, as $\to_{\mathtt{m}}$ and $\to_{\mathtt{e}}$ are closed by applicative contexts.

(c) *Application Right.* It follows by the *i.h.*, as $\to_{\mathtt{m}}$ and $\to_{\mathtt{e}}$ are closed by applicative contexts.

(d) *Step inside a $\beta$-context, i.e.* $t = (\lambda x.s)r \to_{\beta_v^\flat} (\lambda x.q)r = u$ with $s \to_{\beta_v^\flat} q$. By *i.h.*, $s \to_{\mathsf{m}}^{+}\to_{\mathsf{e}} p \;{}_{\mathsf{m}}^{*}\!\!\leftarrow q$. Now, $\to_{\mathsf{m}}$ and $\to_{\mathsf{e}}$ are not closed by balanced contexts, but it is enough to apply a further $\to_{\mathsf{m}}$ step to the balanced context (as $\to_{\mathsf{m}}$ and $\to_{\mathsf{e}}$ are instead closed by substitution contexts), obtaining $(\lambda x.s)r \to_{\mathsf{m}} s[x{\leftarrow}r] \to_{\mathsf{m}}^{+}\to_{\mathsf{e}} p[x{\leftarrow}r] \;{}_{\mathsf{m}}^{*}\!\!\leftarrow q[x{\leftarrow}r] \;{}_{\mathsf{m}}\!\!\leftarrow (\lambda x.q)r$. □

See p. 15

**Lemma 14** (Projecting a $\mathtt{sh}$-Step on $\to_{\mathtt{vsub}_\equiv}$ via $\mathsf{m}$-nf)**.** *Let* $t, u \in \Lambda$.

1. *If* $t \to_{\sigma^\flat} u$ *then* $\mathtt{m}(t) \equiv \mathtt{m}(u)$.
2. *If* $t \to_{\beta_v^\flat} u$ *then* $\mathtt{m}(t) \to_{\mathsf{e}}\to_{\mathsf{m}}^{*} \mathtt{m}(u)$.

*Proof.*

1. By Lemma 42.1 there exist $s, r \in \Lambda_{\mathtt{vsub}}$ s.t. $t \to_{\mathsf{m}}^{+} s \equiv r \;{}_{\mathsf{m}}^{+}\!\!\leftarrow u$. By existence and uniqueness of the $\mathsf{m}$-normal form (Propositions 4.1-1 and Prop. 25.1), $s \to_{\mathsf{m}}^{+} \mathtt{m}(s) = \mathtt{m}(t)$. By Lemma 5.2, there is $q \in \Lambda_{\mathtt{vsub}}$ s.t. $r \to_{\mathsf{m}}^{+} q \equiv \mathtt{m}(t)$. By Lemma 5.3, $q$ is $\mathsf{m}$-normal; in particular, $q = \mathtt{m}(r) = \mathtt{m}(u)$ according to Prop. 25.1. Thus, $\mathtt{m}(t) \equiv q = \mathtt{m}(u)$.
2. By Lemma 42.2 there are $s, r \in \Lambda_{\mathtt{vsub}}$ such that $t \to_{\mathsf{m}}^{+} s \to_{\mathsf{e}} r \;{}_{\mathsf{m}}^{*}\!\!\leftarrow u$. By existence and uniqueness of the $\mathsf{m}$-normal form (Propositions 4.1-1 and Prop. 25.1), $\mathtt{m}(s) = \mathtt{m}(t)$. As $\mathtt{m}(t) \;{}_{\mathsf{m}}^{*}\!\!\leftarrow s \to_{\mathsf{e}} r$, there is $q \in \Lambda_{\mathtt{vsub}}$ s.t. $\mathtt{m}(t) \to_{\mathsf{e}} q \;{}_{\mathsf{m}}^{*}\!\!\leftarrow r$ according to strong commutation of $\to_{\mathsf{m}}$ and $\to_{\mathsf{e}}$ (Prop. 4.2). Thus, $\mathtt{m}(t) \to_{\mathsf{e}} q \;{}_{\mathsf{m}}^{*}\!\!\leftarrow u$ and hence $\mathtt{m}(t) \to_{\mathsf{e}}\to_{\mathsf{m}}^{*} \mathtt{m}(u)$ since $\mathtt{m}(u) = \mathtt{m}(q)$ by Prop. 25.1.

□

See p. 15

**Lemma 15** (Projection Preserves Normal Forms)**.** *Let* $t \in \Lambda$. *If* $t$ *is* $\mathtt{sh}$*-normal then* $\mathtt{m}(t)$ *is* $\mathtt{vsub}$*-normal.*

*Proof.* In [8, Prop. 12] (where the reduction $\to_{\mathtt{sh}}$ is denoted by $\to_{\mathtt{w}}$) it has been shown that:

1. a term is $\mathtt{sh}$-normal iff it is of the form $w$,
2. a term is $\mathtt{sh}$-normal and is neither a value nor a $\beta$-redex (*i.e.* of the form $(\lambda x.t)u$) iff it is of the form $a$,

where the forms $w$ and $a$ are defined by mutual induction as follows:

$$a ::= xv \mid xa \mid aw \qquad\qquad w ::= v \mid a \mid (\lambda x.w)a.$$

The idea is the following: on the one hand, not only terms of the form $a$ are not values but also they cannot reduce to value through $\mathsf{m}$-derivations; on the other hand, any $\mathsf{m}$-derivation from a term of the form $w$ cannot create an ES of the form $[x{\leftarrow}L\langle v\rangle]$, therefore the $\mathsf{e}$-normality of $w$ (which is without ES) is preserved in its $\mathsf{m}$-normal form $\mathtt{m}(w)$ and hence $\mathtt{m}(w)$ is $\mathtt{vsub}$-normal.

More formally, consider the types $a_{\mathtt{vsub}}$ and $w_{\mathtt{vsub}}$ of $\mathtt{vsub}$-terms defined by mutual induction as follows ($v$ is a value, without ES):

$$a_{\mathtt{vsub}} ::= xv \mid xa_{\mathtt{vsub}} \mid a_{\mathtt{vsub}}w_{\mathtt{vsub}}$$
$$w_{\mathtt{vsub}} ::= v \mid a_{\mathtt{vsub}} \mid w_{\mathtt{vsub}}[x{\leftarrow}a_{\mathtt{vsub}}].$$

First, we prove by mutual induction on $a$ and $w$ that the $\mathsf{m}$-normal form $\mathtt{m}(a)$ of $a$ is of the form $a_{\mathtt{vsub}}$, and the $\mathsf{m}$-normal form $\mathtt{m}(w)$ of $w$ is of the form $w_{\mathtt{vsub}}$. The base cases are $\mathtt{m}(v) = v$ (since $\to_{\mathsf{m}}$ does not reduce under $\lambda$'s) and $\mathtt{m}(xv) = xv$. Inductive cases:

1. $\mathtt{m}(xa) = x\mathtt{m}(a) = xa_{\mathtt{vsub}}$ where $\mathtt{m}(a) = a_{\mathtt{vsub}}$ by *i.h.*,
2. $\mathtt{m}(aw) = \mathtt{m}(a)\mathtt{m}(w) = a_{\mathtt{vsub}}w_{\mathtt{vsub}}$ (since $a_{\mathtt{vsub}}$ is not an abstraction) where $\mathtt{m}(a) = a_{\mathtt{vsub}}$ and $\mathtt{m}(w) = w_{\mathtt{vsub}}$ by *i.h.*,
3. $\mathtt{m}((\lambda x.w)a) = \mathtt{m}(w)[x{\leftarrow}\mathtt{m}(a)] = w_{\mathtt{vsub}}[x{\leftarrow}a_{\mathtt{vsub}}]$ (since $a_{\mathtt{vsub}}$ is not of the form $L\langle v\rangle$) where $\mathtt{m}(a) = a_{\mathtt{vsub}}$ and $\mathtt{m}(w) = w_{\mathtt{vsub}}$ by *i.h.*.

To conclude the proof of Lemma 15, it is sufficient to observe that all terms of type $w_{\mathtt{vsub}}$ are $\mathtt{vsub}$-normal, see [3, Lemma 5] (where $\to_{\mathtt{vsub}}$ is denoted by $\to_{\mathtt{w}}$). □

**Theorem 16** (Quantitative Simulation of $\lambda_{\mathtt{sh}}$ in $\lambda_{\mathtt{vsub}}$)**.** *Let $t, u \in \Lambda$. If $d\colon t \to^*_{\mathtt{sh}} u$ then there are $s \in \Lambda_{\mathtt{vsub}}$ and $e\colon t \to^*_{\mathtt{vsub}} s$ such that* See p. 15

1. Qualitative Relationship*: $s \equiv \mathtt{m}(u)$;*
2. Quantitative Relationship*: $|d|_{\beta_v^\flat} = |e|_{\mathsf{e}}$;*
3. Normal Forms*: if $u$ is $\mathtt{sh}$-normal then $s$, $\mathtt{m}(u)$ are $\mathtt{vsub}$-normal.*

*Proof.* First, by straightforward induction on $|d|_{\mathtt{sh}} \in \mathbb{N}$ using the projection via $\mathsf{m}$-normal forms (Lemmas 14.1-2), one proves that there is $e_1 : \mathtt{m}(t) \to^*_{\mathtt{vsub}_{\equiv}} \mathtt{m}(u)$ with $|e_1|_{\mathsf{e}} = |d|_{\beta_v^\flat}$. By postponement of $\equiv$ (Lemma 5.2), there is $e_2 : \mathtt{m}(t) \to^*_{\mathtt{vsub}} \equiv \mathtt{m}(u)$ with $|e_2|_{\mathsf{e}} = |e_1|_{\mathsf{e}}$. Clearly, $t \to^*_{\mathsf{m}} \mathtt{m}(t)$. It easy to check that $s \equiv r$ implies $s \not\to_{\mathsf{e}} r$ for all $s, r \in \Lambda_{\mathtt{vsub}}$. Therefore, there exist $s \in \Lambda_{\mathtt{vsub}}$ and $e\colon t \to^*_{\mathtt{vsub}} s$ such that $s \equiv \mathtt{m}(u)$ and $|e|_{\mathsf{e}} = |e_2|_{\mathsf{e}} = |d|_{\beta_v^\flat}$.

Finally, if moreover $u$ is $\mathtt{sh}$-normal then, since normal forms are preserved by multiplicative projection (Lemma 15), $\mathtt{m}(u)$ is $\mathtt{vsub}$-normal, and hence so is $s$ (Lemma 5.3, because $s \equiv \mathtt{m}(u)$). □

**Corollary 17** (Termination Equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{sh}}$)**.** *Let $t \in \Lambda$. There is a $\mathtt{sh}$-normalizing derivation $d$ from $t$ iff there is a $\mathtt{vsub}$-normalizing derivation $e$ from $t$. Moreover, $|d|_{\beta_v^\flat} = |e|_{\mathsf{e}}$.* See p. 15

*Proof.*

**⇒:** Let $d\colon t \to^*_{\mathtt{sh}} u$ be a $\mathtt{sh}$-normalizing derivation and $e\colon t \to^*_{\mathtt{vsub}} s$ be its projection in $\lambda_{\mathtt{vsub}}$ with $s$ $\mathtt{vsub}$-normal, according to Thm. 12. Then $e$ is a $\mathtt{vsub}$-normalizing derivation from $t$.

**⇐:** By contradiction, suppose that there is a diverging $\mathtt{sh}$-derivation $d$ from $t$ in $\lambda_{\mathtt{sh}}$. Since $\to_{\sigma^\flat}$ is strongly normalizing (Prop. 6.2), necessarily in $d$ there are infinitely many $\beta_v^\flat$-steps. By Thm. 12, $d$ projects to a $\mathtt{vsub}$-derivation in $\lambda_{\mathtt{vsub}}$ that has as many $\mathsf{e}$-steps as the $\beta_v^\flat$-steps in $\lambda_{\mathtt{sh}}$, absurd.

About the length, we have $|d|_{\beta_v^\flat} = |e|_{\mathsf{e}}$ by Thm. 12.2. □

**Corollary 18** (Number of $\beta_v^\flat$-Steps is Invariant)**.** *All $\mathtt{sh}$-normalizing derivations from $t \in \Lambda$ (if any) have the same number of $\beta_v^\flat$-steps.* See p. 15

*Proof.* Let $d\colon t \to^*_{\mathtt{sh}} u$ and $d'\colon t \to^*_{\mathtt{sh}} u'$ be $\mathtt{sh}$-normalizing. By confluence of $\to_{\mathtt{sh}}$ (Prop. 6.3), $u = u'$. According to Thm. 16, $d$ and $d'$ project, respectively, to two $\mathtt{vsub}$-normalizing derivations $e\colon t \to^*_{\mathtt{vsub}} s \in \Lambda_{\mathtt{vsub}}$ and $e'\colon t \to^*_{\mathtt{vsub}} s' \in \Lambda_{\mathtt{vsub}}$ such that $s \equiv \mathtt{m}(u) \equiv s'$, $|e|_{\mathtt{e}} = |d|_{\beta_v^\flat}$ and $|e'|_{\mathtt{e}} = |d'|_{\beta_v^\flat}$. By Prop. 4.3, $|e|_{\mathtt{e}} = |e'|_{\mathtt{e}}$ and hence $|d|_{\beta_v^\flat} = |d'|_{\beta_v^\flat}$. □

### B.3 Proofs of Section 4 (Quantitative Equivalence of $\boldsymbol{\lambda}_{\mathtt{vsub}}$ and $\boldsymbol{\lambda}_{\mathtt{vseq}}$, via $\boldsymbol{\lambda}_{\mathtt{vsub}_k}$)

*Equivalence of $\lambda_{\mathtt{vsub}_k}$ and $\lambda_{\mathtt{vsub}}$* We first give some more details on $\lambda_{\mathtt{vsub}_k}$.

The kernel $\mathtt{vsub}_k$ of $\mathtt{vsub}$ is the sublanguage of $\mathtt{vsub}$. defined by the following grammar of terms and values (by mutual induction), and evaluation contexts:

$$\begin{array}{rr} \mathtt{vsub}_k\text{-Values} & v ::= x \mid \lambda x.t \\ \mathtt{vsub}_k\text{-Terms} & t,u,s ::= v \mid tv \mid t[x{\leftarrow}u] \\ \mathtt{vsub}_k\text{-Evaluation Contexts} & E ::= \langle\cdot\rangle \mid Ev \mid E[x{\leftarrow}u] \mid t[x{\leftarrow}E] \end{array}$$

The rewriting rules are the same of $\mathtt{vsub}$. Note that evaluation contexts of $\mathtt{vsub}_k$ no longer include the case $tE$, because in $\mathtt{vsub}_k$ such contexts cannot surround redexes, as $E$ necessary is the empty context, that can only be filled in with a value, $v$, and values are normal forms. The set of terms of $\mathtt{vsub}_k$ is denoted by $\Lambda_{\mathtt{vsub}_k}$. The restriction of $\to_{\mathtt{vsub}}$ to $\Lambda_{\mathtt{vsub}_k}$ (*i.e.* the closure of $\mapsto_{\mathtt{m}} \cup \mapsto_{\mathtt{e}}$ under $\mathtt{vsub}_k$-evaluation contexts) is denoted by $\to_{\mathtt{vsub}_k}$. Note that $\mathtt{vsub}_k$-terms are closed under substitution of $\mathtt{vsub}_k$-values (*i.e.* if $t$ is a $\mathtt{vsub}_k$-term and $v$ is a $\mathtt{vsub}_k$-value, then $t\{x{\leftarrow}v\}$ is a $\mathtt{vsub}_k$-term), therefore for every $\mathtt{vsub}_k$-term $t$, if $t \to_{\mathtt{vsub}} u$ then $u$ is a $\mathtt{vsub}_k$-term: so, $\to_{\mathtt{vsub}_k}$ is a binary relation on $\Lambda_{\mathtt{vsub}_k}$.

**Lemma 43 (Substitution).** *For any $\mathtt{vsub}$-term $t$ and any $\mathtt{vsub}$-value $v$, one has $t\{x{\leftarrow}v\}^+ = t^+\{x{\leftarrow}v^+\}$.*

*Proof.* By induction on $t$. Cases:

- $t = x$: then, $t\{x{\leftarrow}v\} = v$ and $t^+ = x$, thus $t\{x{\leftarrow}v\}^+ = v^+ = t^+\{x{\leftarrow}v^+\}$.
- $t = y \neq x$: then, $t\{x{\leftarrow}v\} = y$ and $t^+ = y$, hence $t\{x{\leftarrow}v\}^+ = y = t^+\{x{\leftarrow}v^+\}$.
- $t = \lambda y.s$: we can suppose without loss of generality that $y \notin \mathtt{fv}(v) \cup \{x\}$, whence $y \notin \mathtt{fv}(v^+)$. By *i.h.*, $s\{x{\leftarrow}v\}^+ = s^+\{x{\leftarrow}v^+\}$. So, $t\{x{\leftarrow}v\}^+ = \lambda y.s\{x{\leftarrow}v\}^+ = \lambda y.s^+\{x{\leftarrow}v^+\} = t^+\{x{\leftarrow}v^+\}$.
- $t = sr$: then, $t^+ = (s^+y)[y{\leftarrow}s^+]$ with $y \notin \mathtt{fv}(s)$, and we can suppose without loss of generality that $y \notin \mathtt{fv}(v) \cup \{x\}$. By *i.h.*, $s\{x{\leftarrow}v\}^+ = s^+\{x{\leftarrow}v^+\}$ and $r\{x{\leftarrow}v\}^+ = r^+\{x{\leftarrow}v^+\}$, hence $t\{x{\leftarrow}v\}^+ = (s\{x{\leftarrow}v\}^+y)[y{\leftarrow}r\{x{\leftarrow}v\}^+] = (s^+\{x{\leftarrow}v^+\}y)[y{\leftarrow}r^+\{x{\leftarrow}v^+\}] = t^+\{x{\leftarrow}v^+\}$, since $y \neq x$.
- $t = s[y{\leftarrow}r]$: we can suppose without loss of generality that $y \notin \mathtt{fv}(v) \cup \{x\}$. By *i.h.*, $s\{x{\leftarrow}v\}^+ = s^+\{x{\leftarrow}v^+\}$ and $r\{x{\leftarrow}v\}^+ = r^+\{x{\leftarrow}v^+\}$, so $t\{x{\leftarrow}v\}^+ = s\{x{\leftarrow}v\}^+[y{\leftarrow}r\{x{\leftarrow}v\}^+] = s^+\{x{\leftarrow}v^+\}[y{\leftarrow}r^+\{x{\leftarrow}v^+\}] = t^+\{x{\leftarrow}v^+\}$. □

See p. 16

**Lemma 19** (Simulation)**.** *Let $t, u \in \Lambda_{\mathtt{vsub}}$.*

1. Multiplicative*: if $t \to_{\mathtt{m}} u$ then $t^+ \to_{\mathtt{m}} u^+$ or $t^+ \to_{\mathtt{m}}\to_{\mathtt{e}_y}\equiv u^+$;*

2. Exponential Abstractions & Variables*: if* $t \to_{\mathsf{e}_\lambda} u$ *then* $t^+ \to_{\mathsf{e}_\lambda} u^+$*, and if* $t \to_{\mathsf{e}_y} u$ *then* $t^+ \to_{\mathsf{e}_y} u^+$*.*
3. Structural Equivalence*:* $t \equiv u$ *implies* $t^+ \equiv u^+$*.*

*Proof.* Define $L^+$ by $\langle\cdot\rangle^+ := \langle\cdot\rangle$ and $L[x{\leftarrow}r]^+ := L^+[x{\leftarrow}r^+]$. Then note that if $t = L\langle s\rangle$ we have $t^+ = L^+\langle s^+\rangle$. By induction on the evaluation context in which the step takes place. The base cases:

1. *Multiplicative*, *i.e.* $t = L\langle \lambda x.s\rangle r \to_{\mathsf{m}} L\langle s[x{\leftarrow}r]\rangle = u$. Then

$$\begin{array}{rl} t^+ & = (L\langle \lambda x.s\rangle r)^+ \\ & = (L^+\langle \lambda x.s^+\rangle y)[y{\leftarrow}r^+] \\ & \to_{\mathsf{m}} L^+\langle s^+[x{\leftarrow}y]\rangle[y{\leftarrow}r^+] \\ & \to_{\mathsf{e}_y} L^+\langle s^+\{x{\leftarrow}y\}\rangle[y{\leftarrow}r^+] \\ & \equiv L^+\langle s^+\{x{\leftarrow}y\}[y{\leftarrow}r^+]\rangle \\ & =_\alpha L^+\langle s^+[x{\leftarrow}r^+]\rangle \\ & = L\langle s[x{\leftarrow}r]\rangle^+ \qquad = u^+ \end{array}$$

2. *Exponential Abstractions & Variables*, *i.e.* $t = s[x{\leftarrow}L\langle v\rangle] \to_{\mathsf{e}} L\langle s\{x{\leftarrow}v\}\rangle = u$.

$$\begin{array}{rl} t^+ & = s[x{\leftarrow}L\langle v\rangle]^+ \\ & = s^+[x{\leftarrow}L^+\langle v^+\rangle] \\ & \to_{\mathsf{e}} L^+\langle s^+\{x{\leftarrow}v^+\}\rangle \\ & =_{L.43} L^+\langle s\{x{\leftarrow}v\}^+\rangle \\ & = L\langle s\{x{\leftarrow}v\}\rangle^+ \qquad = u^+ \end{array}$$

   Note that the translation $\cdot^+$ maps $\to_{\mathsf{e}_\lambda}$ steps on $\to_{\mathsf{e}_\lambda}$ steps and $\to_{\mathsf{e}_y}$ steps to $\to_{\mathsf{e}_y}$ steps because it maps variables to variables and abstractions to abstractions.
3. *Structural Equivalence*: it is enough to prove that the statement holds for the axioms of $\equiv$, *i.e.* if $t \equiv_{\mathsf{r}} u$ for some $\mathsf{r} \in \{\mathrm{com}, @_{\mathrm{l}}, @_{\mathrm{r}}, [\cdot]\}$, then $t^+ \equiv u^+$. Cases (recall that $\mathtt{vsub}_k$ is a sublanguage of $\mathtt{vsub}$):
   - $t[y{\leftarrow}s][x{\leftarrow}u] \equiv_{\mathrm{com}} t[x{\leftarrow}u][y{\leftarrow}s]$ with $y \notin \mathtt{fv}(u)$ and $x \notin \mathtt{fv}(s)$: then, $t[y{\leftarrow}s][x{\leftarrow}u]^+ = t^+[y{\leftarrow}s^+][x{\leftarrow}u^+] \equiv_{\mathrm{com}} t^+[x{\leftarrow}u^+][y{\leftarrow}s^+] = t[x{\leftarrow}u][y{\leftarrow}s]^+$, since $y \notin \mathtt{fv}(u^+)$ and $x \notin \mathtt{fv}(s^+)$.
   - $t\,s[x{\leftarrow}u] \equiv_{@\mathrm{r}} (ts)[x{\leftarrow}u]$ with $x \notin \mathtt{fv}(t)$: then, $x \notin \mathtt{fv}(t^+)$ and $(ts)^+ = (t^+y)[y{\leftarrow}s^+]$ with $y \notin \mathtt{fv}(t) = \mathtt{fv}(t^+)$, and we can suppose without loss of generality that $y \neq x$. So, $(t\,s[x{\leftarrow}u])^+ = (t^+y)[y{\leftarrow}s^+[x{\leftarrow}u^+]] \equiv_{[\cdot]} (t^+y)[y{\leftarrow}s^+][x{\leftarrow}u^+] = (ts)[x{\leftarrow}u]^+$.
   - $t[x{\leftarrow}u]s \equiv_{@\mathrm{l}} (ts)[x{\leftarrow}u]$ with $x \notin \mathtt{fv}(s)$: then, $x \notin \mathtt{fv}(s^+)$ and $(ts)^+ = (t^+y)[y{\leftarrow}s^+]$ with $y \notin \mathtt{fv}(t) = \mathtt{fv}(t^+)$. We can suppose without loss of generality that $y \notin \mathtt{fv}(u) \cup \{x\}$, hence $t^+[x{\leftarrow}u^+]y \equiv_{@\mathrm{l}} (t^+y)[x{\leftarrow}u^+]$. Therefore, $(t[x{\leftarrow}u]s)^+ = (t^+[x{\leftarrow}u^+]y)[y{\leftarrow}s^+] \equiv (t^+y)[x{\leftarrow}u^+][y{\leftarrow}s^+] \equiv_{[\cdot]} (t^+y)[y{\leftarrow}s^+][x{\leftarrow}u^+] = (ts)[x{\leftarrow}u]^+$.
   - $t[x{\leftarrow}u[y{\leftarrow}s]] \equiv_{[\cdot]} t[x{\leftarrow}u][y{\leftarrow}s]$ with $y \notin \mathtt{fv}(t)$: then, $t[x{\leftarrow}u[y{\leftarrow}s]]^+ = t^+[x{\leftarrow}u^+[y{\leftarrow}s^+]] \equiv_{[\cdot]} t^+[x{\leftarrow}u^+][y{\leftarrow}s^+] = t[x{\leftarrow}u][y{\leftarrow}s]^+$, since $y \notin \mathtt{fv}(t^+)$.

The inductive cases simply follow from the *i.h.*, note indeed that evaluation contexts are translated to evaluation contexts. □

**Theorem 20** (Quantitative Simulation of $\lambda_{\mathtt{vsub}}$ in $\lambda_{\mathtt{vsub}_k}$)**.** *Let $t, u \in \Lambda_{\mathtt{vsub}}$. If $d\colon t \to^*_{\mathtt{vsub}} u$ then there are $s \in \Lambda_{\mathtt{vsub}_k}$ and $e\colon t^+ \to^*_{\mathtt{vsub}_k} s$ such that* See p. 16

1. Qualitative Relationship*: $s \equiv u^+$;*
2. Quantitative Relationship*:*
   1. Multiplicative Steps*: $|e|_{\mathtt{m}} = |d|_{\mathtt{m}}$;*
   2. Exponential Steps*: $|e|_{\mathtt{e}_\lambda} = |d|_{\mathtt{e}_\lambda}$ and $|e|_{\mathtt{e}_y} = |d|_{\mathtt{e}_y} + |d|_{\mathtt{m}}$;*
3. Normal Form*: if $u$ is $\mathtt{vsub}$-normal then $s$ is $\mathtt{m}$-normal and $\mathtt{e}(s)$ is $\mathtt{vsub}_k$-normal.*

*Proof.*

1-2. By induction on $|d|$ using Lemma 19.1 and Lemma 19.2 plus the postponement of $\equiv$ (Lemma 5.1).

3. We prove the following property, noted (*): *if $u$ is an abstraction / variable / compound inert term then $u^+$ is $\mathtt{m}$-normal and $\mathtt{e}(u^+)$ is an abstraction / variable / $\mathtt{vsub}_k$-normal term that is not a value in a substitution context.* The statement then follows from the properties of $\equiv$, because:
   (a) Every normal form $u$ is $\equiv$-equivalent to a clean normal form $u'$;
   (b) Property (*) implies that $u'^+$ is $\mathtt{m}$-normal and $\mathtt{e}(u'^+)$ is $\mathtt{vsub}_k$-normal .
   (c) By the properties of $\equiv$ we have that $u^+ \equiv u'^+$ and $\mathtt{e}(u^+) \equiv \mathtt{e}(u'^+)$ with the same (lack of) redexes.

   By induction on $u$. Cases:
   - *Variable and Abstraction.* Straightforward, as the translation maps variables / abstractions to variables / abstractions.
   - *Compound Inert Term, i.e. $u = if$.* We have $(if)^+ = (i^+x)[x{\leftarrow}f^+]$ with $x$ fresh. By *i.h.*, $i^+$ and $f^+$ are $\mathtt{m}$-normal, so that $u^+$ is $\mathtt{m}$-normal. By *i.h.*, $\mathtt{e}(i^+)$ is $\mathtt{vsub}_k$-normal term that is not an abstraction in a substitution context and $\mathtt{e}(f^+)$ is $\mathtt{vsub}_k$-normal. Now, $u^+ = (i^+x)[x{\leftarrow}f^+] \to^*_{\mathtt{e}} (\mathtt{e}(i^+)x)[x{\leftarrow}\mathtt{e}(f^+)]$. If $f$ is a compound inert term then by *i.h.* $\mathtt{e}(f^+)$ is not a value in a substitution context, and so $(\mathtt{e}(i^+)x)[x{\leftarrow}\mathtt{e}(f^+)]$ is $\mathtt{vsub}_k$-normal. Otherwise, $\mathtt{e}(f^+) = L\langle v\rangle$ and so $(\mathtt{e}(i^+)x)[x{\leftarrow}\mathtt{e}(f^+)] = (\mathtt{e}(i^+)x)[x{\leftarrow}L\langle v\rangle] \to_{\mathtt{e}} L\langle \mathtt{e}(i^+)v\rangle$, that is readily seen to be $\mathtt{vsub}_k$-normal.

□

See p. 16 **Corollary 21** (Linear Termination Equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{vsub}_k}$)**.** *Let $t \in \Lambda_{\mathtt{vsub}}$. There exists a $\mathtt{vsub}$-normalizing derivation $d$ from $t$ iff there exists a $\mathtt{vsub}_k$-normalizing derivation $f$ from $t^+$. Moreover, $|d|_{\mathtt{vsub}} \le |f|_{\mathtt{vsub}_k} \le 3|d|_{\mathtt{vsub}}$.*

*Proof.*

**⇒:** Let $d\colon t \to^*_{\mathtt{vsub}} u$ be a $\mathtt{vsub}$-normalizing derivation and $e\colon t^+ \to^*_{\mathtt{vsub}_k} s$ be its projection in $\lambda_{\mathtt{vsub}_k}$, according to Thm. 20. By Thm. 20.3, the derivation $f$ obtained by extending $e$ with a normalization with respect to $\to_{\mathtt{e}}$ (that always terminate) is a $\mathtt{vsub}_k$-normalizing derivation from $t^+$.

**⇐:** By contradiction, suppose that there is a diverging $\mathtt{vsub}$-derivation from $t$ in $\lambda_{\mathtt{vsub}}$. By Thm. 20 it projects to a $\mathtt{vsub}_k$-derivation from $t^+$ in $\lambda_{\mathtt{vsub}_k}$ that is at least as long as the one in $\lambda_{\mathtt{vsub}}$, absurd.

About lengths, $|e|_{\mathtt{vsub}_k} = |e|_{\mathsf{m}} + |e|_{\mathsf{e}_\lambda} + |e|_{\mathsf{e}_y} =_{Thm.\ 20.2} 2|d|_{\mathsf{m}} + |d|_{\mathsf{e}_\lambda} + |d|_{\mathsf{e}_y} = |d|_{\mathtt{vsub}} + |d|_{\mathsf{m}}$. Clearly, $|d|_{\mathtt{vsub}} \leq |d|_{\mathtt{vsub}} + |d|_{\mathsf{m}} = |e|_{\mathtt{vsub}_k} \leq |f|_{\mathtt{vsub}_k}$. Again by Thm. 20.2, the further exponential normalization needed to reach a $\mathtt{vsub}_k$-normal form is bounded by $|e|_{\mathsf{e}_\lambda} + |e|_{\mathsf{e}_y} = |d|_{\mathsf{e}_\lambda} + |d|_{\mathsf{m}} + |d|_{\mathsf{e}_y} = |d|_{\mathtt{vsub}}$. Summing up, $|d|_{\mathtt{vsub}} \leq |f|_{\mathtt{vsub}_k} \leq |d|_{\mathtt{vsub}} + |d|_{\mathsf{m}} + |d|_{\mathtt{vsub}} \leq 3|d|_{\mathtt{vsub}}$. □

*Equivalence of $\lambda_{\mathtt{vsub}_k}$ and $\lambda_{\mathtt{vseq}}$.*

**Lemma 44 (Translation and Substitution Commute).** *Let $v$, $v'$ be values and $t$ be a term of $\mathtt{vsub}_k$.*

1. Values*:* $(v'\{x{\leftarrow}v\})^\bullet = v'^\bullet\{x{\leftarrow}v^\bullet\}$*;*
2. Terms*:* $\underline{t\{x{\leftarrow}v\}} = \underline{t}\{x{\leftarrow}v^\bullet\}$*.*

*Proof.*

1. Cases:
   - *Variable, i.e.* $t = x$. Then $(x\{x{\leftarrow}v\})^\bullet = v^\bullet = x\{x{\leftarrow}v^\bullet\} = x^\bullet\{x{\leftarrow}v^\bullet\}$.
   - *Abstraction, i.e.* $t = \lambda y.u$.

$$\begin{array}{rcl} ((\lambda y.u)\{x{\leftarrow}v\})^\bullet & = & (\lambda y.u\{x{\leftarrow}v\})^\bullet \\ & = & \lambda y.\underline{u\{x{\leftarrow}v\}} \\ & =_{P.2} & \lambda y.\underline{u}\{x{\leftarrow}v^\bullet\} \\ & = & (\lambda y.\underline{u})\{x{\leftarrow}v^\bullet\} = (\lambda y.u)^\bullet\{x{\leftarrow}v^\bullet\} \end{array}$$

2. By induction on $t$. Cases:
   - *Value, i.e.* $t = v'$. Note that $v'\{x{\leftarrow}v\}$ is a value. Then $\underline{v'\{x{\leftarrow}v\}} = \langle (v'\{x{\leftarrow}v\})^\bullet \,|\, \epsilon \rangle =_{P.1} \langle v'^\bullet\{x{\leftarrow}v^\bullet\} \,|\, \epsilon \rangle = \langle v'^\bullet \,|\, \epsilon \rangle\{x{\leftarrow}v^\bullet\} = \underline{v'}\{x{\leftarrow}v^\bullet\}$.
   - *Application, i.e.* $t = uv'$.

$$\begin{array}{rcll} \underline{uv'\{x{\leftarrow}v\}} & = & \underline{u\{x{\leftarrow}v\}v'\{x{\leftarrow}v\}} & \\ & = & \underline{u\{x{\leftarrow}v\}}@(v'\{x{\leftarrow}v\}^\bullet \cdot \epsilon) & \\ & =_{i.h.} & \underline{u}\{x{\leftarrow}v^\bullet\}@(v'\{x{\leftarrow}v\}^\bullet \cdot \epsilon) & \\ & =_{P.1} & \underline{u}\{x{\leftarrow}v^\bullet\}@(v'^\bullet\{x{\leftarrow}v^\bullet\} \cdot \epsilon) & \\ & = & \underline{u}@(v'^\bullet \cdot \epsilon)\{x{\leftarrow}v^\bullet\} & = \underline{uv'}\{x{\leftarrow}v^\bullet\} \end{array}$$

   - *Substitution, i.e.* $t = u[y{\leftarrow}s]$.

$$\begin{array}{rcll} \underline{u[y{\leftarrow}s]\{x{\leftarrow}v\}} & = & \underline{u\{x{\leftarrow}v\}[y{\leftarrow}s\{x{\leftarrow}v\}]} & \\ & = & \underline{s\{x{\leftarrow}v\}}@\tilde{\mu}y.\underline{u\{x{\leftarrow}v\}} & \\ & =_{i.h.} & \underline{s}\{x{\leftarrow}v^\bullet\}@\tilde{\mu}y.\underline{u}\{x{\leftarrow}v^\bullet\} & \\ & = & (\underline{s}@\tilde{\mu}y.\underline{u})\{x{\leftarrow}v^\bullet\} & = \underline{u[y{\leftarrow}s]}\{x{\leftarrow}v^\bullet\} \end{array}$$

□

**Lemma 45.** *Let $t$ be a $\mathtt{vsub}_k$-term. Then there exist a command evaluation context $C$ and a environment evaluation context $D$ such that $\underline{t} = C\langle D\langle\epsilon\rangle\rangle$.*

*Proof.* By induction on $t$. Cases:

1. *Variable, i.e.* $t = x$. Trivial just take $C \coloneqq \langle\cdot\rangle$ and $D \coloneqq \langle x \,|\, \langle\cdot\rangle\rangle$.
2. *Abstraction, i.e.* $t = \lambda x.u$. Trivial just take $C \coloneqq \langle\cdot\rangle$ and $D \coloneqq \langle \lambda x.\underline{u} \,|\, \langle\cdot\rangle\rangle$.
3. *Application, i.e.* $t = uv$.

$$\begin{array}{rcl} \underline{uv} & = & \underline{u}@(v^\bullet \cdot \epsilon) \\ & =_{i.h.} & C'\langle D'\langle\epsilon\rangle\rangle@(v^\bullet \cdot \epsilon) \\ & = & C'\langle D'\langle v^\bullet \cdot \epsilon\rangle\rangle \end{array}$$

The statement holds with respect to $C \coloneqq C'$ and $D \coloneqq D'\langle x \cdot \langle\cdot\rangle\rangle$.

4. *Substitution, i.e.* $t = u[x{\leftarrow}s]$.

$$\begin{array}{rcl} \underline{u[x{\leftarrow}s]} & = & \underline{s}@\tilde{\mu}x.\underline{u} \\ & =_{i.h.} & C'\langle D'\langle\epsilon\rangle\rangle@\tilde{\mu}x.\underline{u} \\ & = & C'\langle D'\langle\tilde{\mu}x.\underline{u}\rangle\rangle \\ & =_{i.h.} & C'\langle D'\langle\tilde{\mu}x.C''\langle D''\langle\epsilon\rangle\rangle\rangle\rangle \end{array}$$

The statement holds with respect to $C \coloneqq C'\langle D'\langle\tilde{\mu}x.C''\rangle\rangle$ and $D \coloneqq D''$. □

**Lemma 46.** *Let $L$ be a substitution context of* $\mathtt{vsub}$*. There exists a command evalution context $C$ such that $\underline{L\langle t\rangle} = C\langle\underline{t}\rangle$ for any $\mathtt{vsub}_k$-term $t$. Moreover, $\mathtt{fv}(C) = \mathtt{fv}(L)$ and $C$ and $L$ capture they same variables of $t$.*

*Proof.* By induction on $L$. Cases:

1. *Empty Context, i.e.* $L = \langle\cdot\rangle$. Just take $C \coloneqq \langle\cdot\rangle$.
2. *Non-Empty Context, i.e.* $L = L'[x{\leftarrow}u]$. Then

$$\begin{array}{rcl} \underline{L'\langle t\rangle[x{\leftarrow}u]} & = & \underline{u}@\tilde{\mu}x.\underline{L'\langle t\rangle} \\ & =_{i.h.} & \underline{u}@\tilde{\mu}x.C'\langle\underline{t}\rangle \\ & =_{L.45} & C''\langle D\langle\epsilon\rangle\rangle@\tilde{\mu}x.C'\langle\underline{t}\rangle \\ & = & C''\langle D\langle\tilde{\mu}x.C'\langle\underline{t}\rangle\rangle\rangle \end{array}$$

The statement holds with respect to $C \coloneqq C''\langle D\langle\tilde{\mu}x.C'\rangle\rangle$. The *moreover* part follows from the *moreover* part of Lemma 45 and the *i.h.* □

See p. 17

**Lemma 22** (Simulation of $\to_{\mathtt{vsub}_k}$ by $\to_{\mathtt{vseq}}$)**.** *Let $t, u \in \Lambda_{\mathtt{vsub}_k}$.*

1. Multiplicative*: if $t \to_{\mathtt{m}} u$ then $\underline{t} \to_{\bar{\lambda}} \underline{u}$.*
2. Exponential*: if $t \to_{\mathtt{e}} u$ then $\underline{t} \to_{\tilde{\mu}} \underline{u}$.*

*Proof.* Both points are proved by induction on the evaluation context $E$ in which the step takes place. Cases:

– *Root case, i.e.* $E = \langle\cdot\rangle$.
  1. *Multiplicative Step*: $t = L\langle\lambda x.s\rangle v \mapsto_{\mathtt{m}} L\langle s[x{\leftarrow}v]\rangle = u$.

$$\begin{array}{rcl} \underline{L\langle\lambda x.s\rangle v} & = & \underline{L\langle\lambda x.s\rangle}@(v^\bullet \cdot \epsilon) \\ & =_{L.46} & C\langle\underline{\lambda x.s}\rangle@(v^\bullet \cdot \epsilon) \\ & = & C\langle\langle\lambda x.\underline{s} \,|\, \epsilon\rangle\rangle@(v^\bullet \cdot \epsilon) \\ & = & C\langle\langle\lambda x.\underline{s} \,|\, \epsilon\rangle@(v^\bullet \cdot \epsilon)\rangle \\ & = & C\langle\langle\lambda x.\underline{s} \,|\, v^\bullet \cdot \epsilon\rangle\rangle \\ & \to_{\bar{\lambda}} & C\langle\langle v^\bullet \,|\, \tilde{\mu}x.\underline{s}@\epsilon\rangle\rangle \\ & = & C\langle\langle v^\bullet \,|\, \tilde{\mu}x.\underline{s}\rangle\rangle \\ & = & C\langle\underline{s[x{\leftarrow}v]}\rangle \quad =_{L.46} \underline{L\langle s[x{\leftarrow}v]\rangle} \end{array}$$

2. *Exponential Step*: $t = s[x{\leftarrow}L\langle v\rangle] \mapsto_{\mathsf{e}} L\langle s\{x{\leftarrow}v\}\rangle = u$.

$$\begin{array}{rcl} \underline{s[x{\leftarrow}L\langle v\rangle]} & = & \underline{L\langle v\rangle}@\tilde{\mu}x.\underline{s} \\ & =_{L.46} & C\langle\underline{v}\rangle@\tilde{\mu}x.\underline{s} \\ & = & C\langle\langle v^\bullet \,|\, \epsilon\rangle\rangle@\tilde{\mu}x.\underline{s} \\ & = & C\langle\langle v^\bullet \,|\, \tilde{\mu}x.\underline{s}\rangle\rangle \\ & \to_{\tilde{\mu}} & C\langle\underline{s}\{x{\leftarrow}v^\bullet\}\rangle \\ & =_{L.44} & C\langle\underline{s\{x{\leftarrow}v\}}\rangle \qquad =_{L.46} \underline{L\langle s\{x{\leftarrow}v\}\rangle} \end{array}$$

– *Inductive Cases*: for each case the two points differs only in the kind of the rewriting step, so we treat them compactly, by referring to $\to_{\mathtt{vsub}_k}$ and $\to_{\mathtt{vseq}}$
  - *Left Application, i.e.* $t = sv \to_{\mathtt{vsub}_k} rv = u$ with $s \to_{\mathtt{vsub}_k} r$. By *i.h.*, $\underline{s} \to_{\mathtt{vseq}} \underline{r}$. And by Lemma 38.2, $\underline{sv} = \underline{s}@(\underline{v}\cdot\epsilon) \to_{\mathtt{vseq}} \underline{r}@(\underline{v}\cdot\epsilon) = \underline{rv}$.
  - *Left of a Substitution*: *i.e.* $t = s[x{\leftarrow}q] \to_{\mathtt{vsub}_k} r[x{\leftarrow}q] = u$ with $s \to_{\mathtt{vsub}_k} r$. By *i.h.*, $\underline{s} \to_{\mathtt{vseq}} \underline{r}$. By Lemma 37, $\underline{s[x{\leftarrow}q]} = \underline{q}@\tilde{\mu}x.\underline{s} \to_{\mathtt{vseq}} \underline{q}@\tilde{\mu}x.\underline{r} = \underline{s[x{\leftarrow}r]}$.
  - *Inside a Substitution*: *i.e.* $t = s[x{\leftarrow}q] \to_{\mathtt{vsub}_k} s[x{\leftarrow}r] = u$ with $q \to_{\mathtt{vsub}_k} r$. By *i.h.*, $\underline{q} \to_{\mathtt{vseq}} \underline{r}$. And by Lemma 38.2, $\underline{s[x{\leftarrow}q]} = \underline{q}@\tilde{\mu}x.\underline{s} \to_{\mathtt{vseq}} \underline{r}@\tilde{\mu}x.\underline{s} = \underline{s[x{\leftarrow}r]}$. □

**Theorem 23** (Quantitative Simulation of $\lambda_{\mathtt{vsub}_k}$ in $\lambda_{\mathtt{vseq}}$). *Let* $t, u \in \Lambda_{\mathtt{vsub}_k}$. *If* $d\colon t \to^*_{\mathtt{vsub}_k} u$ *then there is* $e\colon \underline{t} \to^*_{\mathtt{vseq}} \underline{u}$ *such that* ($|e|_{\mathtt{vseq}}$ *denotes the length of* $e$) See p. 17

1. Multiplicative Steps*:* $|d|_{\mathsf{m}} = |e|_{\bar{\lambda}}$ *(the number* $\bar{\lambda}$*-steps in* $e$*);*
2. Exponential Steps*:* $|d|_{\mathsf{e}} = |e|_{\tilde{\mu}}$ *(the number* $\tilde{\mu}$*-steps in* $e$*), so* $|d|_{\mathtt{vsub}_k} = |e|_{\mathtt{vseq}}$*;*
3. Normal Form*: if* $u$ *is* $\mathtt{vsub}_k$*-normal then* $\underline{u}$ *is* $\mathtt{vseq}$*-normal.*

*Proof.* The existence of $e$ and the first two points are immediate consequences of Lemma 22. We prove Point 3 by proving that the translation of a clean normal form $t$ of $\lambda_{\mathtt{vsub}_k}$ is normal. Cases of $u$:

– *Value*: then clearly $\underline{u} = \langle u^\bullet \,|\, \epsilon\rangle$ is normal.
– *Compound Inert Term*: then $u$ has the form $u = xv_1 \ldots v_k$. A straightforward induction on $k$ shows that it translates to $\langle x \,|\, v_1^\bullet\cdot\ldots\cdot v_k^\bullet\cdot\epsilon\rangle$, that is normal.
– *Substitution*: then $u$ has the form $u = s[x{\leftarrow}i]$ where $s$ is a clean normal form and $i$ is a compound inert term. If $i = yv_1 \ldots v_k$ then $\underline{i} = \langle y \,|\, v_1^\bullet\cdot\ldots\cdot v_k^\bullet\cdot\epsilon\rangle$ and $\underline{s[x{\leftarrow}i]} = \langle y \,|\, v_1^\bullet\cdot\ldots\cdot v_k^\bullet\cdot\epsilon\rangle@\tilde{\mu}x.\underline{s} = \langle y \,|\, v_1^\bullet\cdot\ldots\cdot v_k^\bullet\cdot\tilde{\mu}x.\underline{s}\rangle$, that is normal because by *i.h.* $\underline{s}$ is normal. □

**Corollary 24** (Linear Termination Equivalence of $\lambda_{\mathtt{vsub}_k}$ and $\lambda_{\mathtt{vseq}}$). *Let* $t \in \Lambda_{\mathtt{vsub}_k}$. *There is a* $\mathtt{vsub}_k$*-normalizing derivation* $d$ *from* $t$ *iff there is a* $\mathtt{vseq}$*-normalizing derivation* $e$ *from* $\underline{t}$. *Moreover,* $|d|_{\mathtt{vsub}_k} = |e|_{\mathtt{vseq}}$, $|d|_{\mathsf{e}} = |e|_{\tilde{\mu}}$ *and* $|d|_{\mathsf{m}} = |e|_{\bar{\lambda}}$. See p. 17

*Proof.* **⇒:** Let $d\colon t \to^*_{\mathtt{vsub}_k} u$ be a $\mathtt{vsub}_k$-normalizing derivation and $e\colon \underline{t} \to^*_{\mathtt{vseq}} \underline{u}$ be its projection in $\lambda_{\mathtt{vseq}}$, according to Thm. 23. Then $e$ is a $\mathtt{vsub}_k$-normalizing derivation from $\underline{t}$, since the $\mathtt{vsub}_k$-normality of $u$ implies the $\mathtt{vseq}$-normality of $\underline{u}$ by Thm. 23.3.

**$\Leftarrow$:** By contradiction, suppose that there is a diverging $\mathtt{vsub}_k$-derivation from $t$ in $\lambda_{\mathtt{vsub}_k}$. By Thm. 23 it projects to a $\mathtt{vseq}$-derivation from $\underline{t}$ in $\lambda_{\mathtt{vseq}}$ that is at least as long as the one in $\lambda_{\mathtt{vsub}_k}$, which is absurd since $\underline{t}$ is $\mathtt{vseq}$-normalizable and all $\mathtt{vseq}$-normalizing derivations from $\underline{t}$ have the same length by Prop. 7.3.

The result about lengths follows immediately from Thm. 23.1-2. □

*Structural equivalence for $\lambda_{\mathtt{vseq}}$.*

*Remark 47.* Every environment evaluation context can be uniquely written as $D = \langle v \,|\, v_1 \cdot \ldots \cdot v_n \cdot e\rangle$ where either $e = \epsilon$ or $e = \tilde{\mu}x.c$.

**Lemma 48.** *Let $t$ and $u$ be $\mathtt{vsub}_k$-terms.*

1. *If $t \equiv_{[\cdot],@\mathrm{l}} t'$ then $\underline{t} = \underline{t'}$.*
2. *If $t \equiv_{\mathrm{com}} t'$ then $\underline{t} \simeq_{\tilde{\mu}\tilde{\mu}} \underline{t'}$.*

*Proof.* 1. If $t \equiv_{[\cdot]} t'$, then $t = s[x{\leftarrow}r[y{\leftarrow}u]]$ and $t' = s[x{\leftarrow}r][y{\leftarrow}u]$. So, just apply the translation and Lemma 38.1 (setting $C = \underline{u}@\tilde{\mu}y.\langle\cdot\rangle$, $c = \underline{r}$ and $e = \tilde{\mu}x.\underline{s}$):

$$\underline{t} = \underline{s[x{\leftarrow}r[y{\leftarrow}u]]} = (\underline{u}@\tilde{\mu}y.\underline{r})@\tilde{\mu}x.\underline{s} = \underline{u}@\tilde{\mu}y.(\underline{r}@\tilde{\mu}x.\underline{s}) = \underline{s[x{\leftarrow}r][y{\leftarrow}u]} = \underline{t'}.$$

If $t \equiv_{@\mathrm{l}} t'$, then $t = s[x{\leftarrow}u]v$ and $t' = (sv)[x{\leftarrow}u]$. So, just apply the translation and Lemma 38.1 (setting $C = \underline{u}@\tilde{\mu}x.\langle\cdot\rangle$, $c = \underline{s}$ and $e = v^{\bullet}\cdot\epsilon$):

$$\underline{t} = \underline{s[x{\leftarrow}u]v} = (\underline{u}@\tilde{\mu}x.\underline{s})@(v^{\bullet}\cdot\epsilon) = \underline{u}@\tilde{\mu}x.(\underline{s}@(v^{\bullet}\cdot\epsilon)) = \underline{(sv)[x{\leftarrow}u]} = \underline{t'}.$$

2. As $t \equiv_{\mathrm{com}} t'$, then $t = s[y{\leftarrow}r][x{\leftarrow}u]$ and $t' = s[x{\leftarrow}u][y{\leftarrow}r]$ with $x \notin \mathtt{fv}(s)$ and $y \notin \mathtt{fv}(r)$. So, just apply the translation and the definition of $\simeq_{\tilde{\mu}\tilde{\mu}}$, the only axiom generating $\simeq$ (setting $D = \underline{u}@\langle\cdot\rangle$ and $D' = \underline{r}@\langle\cdot\rangle$):

$$\underline{t} = \underline{s[y{\leftarrow}r][x{\leftarrow}u]} = u@\tilde{\mu}x.(\underline{r}@\tilde{\mu}y.\underline{s}) \simeq_{\tilde{\mu}\tilde{\mu}} r@\tilde{\mu}y.(\underline{u}@\tilde{\mu}x.\underline{s}) = \underline{s[x{\leftarrow}u][y{\leftarrow}r]} = \underline{t'}.$$

□

**Proposition 49 (Simulation of $\equiv$ by $\simeq$).** *Let $t$ and $t'$ be $\mathtt{vsub}_k$-terms. If $t \equiv t'$ then $\underline{t} \simeq \underline{t'}$.*

*Proof.* First, observe that there are no $u, u' \in \Lambda_{\mathtt{vsub}_k}$ such that $u \equiv_{@\mathrm{r}} u'$: indeed, $u \equiv_{@\mathrm{r}} u'$ implies that $u = s\,r[x{\leftarrow}q]$ and $r[x{\leftarrow}q]$ is not a value, therefore $u \notin \Lambda_{\mathtt{vsub}_k}$.

Let $\equiv'$ the closure of $\equiv_{@\mathrm{l}} \cup \equiv_{\mathrm{com}} \cup \equiv_{[\cdot]}$ under evaluation contexts of $\lambda_{\mathtt{vsub}_k}$. As $\equiv$ on $\Lambda_{\mathtt{vsub}_k}$ is just the reflexive-transitive and symmetric closure of $\equiv$ (and $\simeq$ is an equivalence relation), in order to prove Prop. 49 it is enough to prove that the following statement $(*)$: for every $t, t' \in \Lambda_{\mathtt{vsub}_k}$, if $t \equiv' t'$ then $\underline{t} \simeq \underline{t'}$. The proof of $(*)$ is by induction on the definition $t \equiv' t'$.

The base cases (*i.e.* when $t \equiv_{@\mathrm{l}} t'$ or $t \equiv_{\mathrm{com}} t'$ or $t \equiv_{[\cdot]} t'$) are already proved in Lemma 48. Concerning the inductive cases, we have:

– *Application Left, i.e.* $t \coloneqq uv \equiv u'v =: t'$ with $u \equiv u'$: by *i.h.*, $\underline{u} \simeq \underline{u'}$; so, $\underline{t} = \underline{u}@(v^{\bullet}\cdot\epsilon) \simeq \underline{u'}@(v^{\bullet}\cdot\epsilon) = \underline{t'}$;

- *Left of a Substitution, i.e.* $t := u[x\leftarrow s] \equiv u'[x\leftarrow s] =: t'$ with $u \equiv u'$: by *i.h.*, $\underline{u} \simeq \underline{u'}$; so, $\underline{t} = \underline{s}@\tilde{\mu}x.\underline{u} \simeq \underline{s}@\tilde{\mu}x.\underline{u'} = \underline{t'}$;
- *Inside a Substitution, i.e.* $t := s[x\leftarrow u] \equiv s[x\leftarrow u'] =: t'$ with $u \equiv u'$: by *i.h.*, $\underline{u} \simeq \underline{u'}$; thus, $\underline{t} = \underline{u}@\tilde{\mu}x.\underline{s} \simeq \underline{u'}@\tilde{\mu}x.\underline{s} = \underline{t'}$. □

**Proposition 50 (Basic Properties of Structural Equivalence $\simeq$).** *Let* $c_0, c_1$ *be commands and* $\mathsf{r} \in \{\bar{\lambda}, \tilde{\mu}, \mathtt{vseq}\}$.

1. Strong Bisimulation of $\simeq$ wrt $\to_{\mathtt{vseq}}$*: if* $c_0 \simeq c_1$ *and* $c_1 \to_{\mathsf{r}} c_2$ *then there exists a command* $c_3$ *such that* $c_0 \to_{\mathsf{r}} c_3 \simeq c_2$.
2. Postponement of $\simeq$ wrt $\to_{\mathtt{vseq}}$*: if* $d\colon c_0 \to_{\mathtt{vseq}_{\simeq}}^{*} c_1$ *then there are* $c_2 \simeq c_1$ *and* $e\colon c_0 \to_{\mathtt{vseq}}^{*} c_2$ *such that* $|d|_{\mathtt{vseq}} = |e|_{\mathtt{vseq}}$, $|d|_{\tilde{\mu}} = |e|_{\tilde{\mu}}$ *and* $|d|_{\bar{\lambda}} = |e|_{\bar{\lambda}}$.
3. Normal Forms*: if* $t \simeq u$ *then* $t$ *is* $\mathsf{r}$*-normal iff* $u$ *is* $\mathsf{r}$*-normal.*
4. Strong confluence*:* $\to_{\mathtt{vseq}_{\simeq}}$ *is strongly confluent.*

*Proof.* 1. It is enough to prove the following statement $(*)$: if $c_0 \simeq' c_1$ and $c_1 \to_{\mathsf{r}} c_2$ then there exists a command $c_3$ such that $c_0 \to_{\mathsf{r}} c_3 \simeq' c_2$, where $\simeq'$ is the reflexive closure under command evaluation contexts of $\simeq_{\tilde{\mu}\tilde{\mu}}$, the unique axiom generating the equivalence $\simeq$. Indeed $\simeq'$ is reflexive and symmetric, therefore $\simeq$ is just the transitive closure of $\simeq'$, so the proof of Prop. 50.1 follows immediately.
The proof of $(*)$ is by induction on the definition of $\simeq'$.
In the inductive cases the proof follows immediately from the *i.h.*, since $\simeq'$ and $\to_{\mathsf{r}}$ are closed under the same contexts.
Concerning the base cases, according to Remark 47, we have

$$\begin{aligned} c_0 &:= \langle v \,|\, v_1 \cdot \ldots \cdot v_n \cdot \tilde{\mu}x.\langle v' \,|\, v'_1 \cdot \ldots \cdot v'_{n'} \cdot \tilde{\mu}y.c\rangle\rangle \\ &\simeq_{\tilde{\mu}\tilde{\mu}} \langle v' \,|\, v'_1 \cdot \ldots \cdot v'_{n'} \cdot \tilde{\mu}y.\langle v \,|\, v_1 \cdot \ldots \cdot v_n \cdot \tilde{\mu}x.c\rangle\rangle =: c_1 \end{aligned}$$

where $x \notin \mathtt{fv}(v') \cup \bigcup_{i'=1}^{n'} \mathtt{fv}(v'_{i'})$ and $y \notin \mathtt{fv}(v) \cup \bigcup_{i=1}^{n} \mathtt{fv}(v_i)$. Thus there are only four cases:

(a) *Internal* $\bar{\lambda}$*-step, i.e.* $v = \lambda z.c'$, $n > 0$ and

$$c_1 \to_{\bar{\lambda}} \langle v' \,|\, v'_1 \cdot \ldots \cdot v'_{n'} \cdot \tilde{\mu}y.\langle v_1 \,|\, (\tilde{\mu}z.c')@(v_2 \cdot \ldots \cdot v_n \cdot \tilde{\mu}x.c)\rangle\rangle = c_2$$

then, $c_0 \to_{\bar{\lambda}} \langle v_1 \mid (\tilde{\mu}z.c')@(v_2 \cdot \ldots \cdot v_n \cdot \tilde{\mu}x.\langle v' \mid v'_1 \cdot \ldots \cdot v'_{n'} \cdot \tilde{\mu}y.c\rangle)\rangle \simeq' c_2$, where the last equivalence holds by applying the axiom $\simeq_{\tilde{\mu}\tilde{\mu}}$ with the environment evaluation contexts $D = \langle v_1 \mid (\tilde{\mu}z.c')@v_2 \cdot \ldots \cdot v_n \cdot \langle\cdot\rangle\rangle$ and $D' = \langle v' \,|\, v'_1 \cdot \ldots \cdot v'_{n'} \cdot \langle\cdot\rangle\rangle$.

(b) *External* $\bar{\lambda}$*-step, i.e.* $v' = \lambda z.c'$, $n' > 0$ and

$$c_1 \to_{\bar{\lambda}} \langle v'_1 \,|\, (\tilde{\mu}z.c')@(v'_2 \cdot \ldots \cdot v'_{n'} \cdot \tilde{\mu}y.\langle v \,|\, v_1 \cdot \ldots \cdot v_n \cdot \tilde{\mu}x.c\rangle)\rangle = c_2$$

then, $c_0 \to_{\bar{\lambda}} \langle v \,|\, v_1 \cdot \ldots \cdot v_n \cdot \tilde{\mu}x.\langle v'_1 \,|\, (\tilde{\mu}z.c')@(v'_2 \cdot \ldots \cdot v'_n \cdot \tilde{\mu}y.c)\rangle\rangle \simeq' c_2$, where the last equivalence holds by applying the axiom $\simeq_{\tilde{\mu}\tilde{\mu}}$ with the environment evaluation contexts $D = \langle v \,|\, v_1 \cdot \ldots \cdot v_n \langle\cdot\rangle\rangle$ and $D' = \langle v'_1 \,|\, (\tilde{\mu}z.c')@v'_2 \cdot \ldots \cdot v'_{n'} \cdot \langle\cdot\rangle\rangle$.

(c) *Internal* $\tilde{\mu}$*-step, i.e.* $n = 0$ and $c_1 \to_{\tilde{\mu}} \langle v' \,|\, v'_1 \cdot \ldots \cdot v'_{n'} \cdot \tilde{\mu}y.c\{x\leftarrow v\}\rangle = c_2$: then, $c_0 \to_{\tilde{\mu}} c_2$ since $x \notin \mathtt{fv}(v') \cup \bigcup_{i'=1}^{n'} \mathtt{fv}(v'_{i'})$.

(d) *External $\tilde{\mu}$-step, i.e.* $n' = 0$ and $c_1 \to_{\tilde{\mu}} \langle v \,|\, v_1 \cdot \ldots \cdot v_n \cdot \tilde{\mu}x.c\{y{\leftarrow}v'\}\rangle = c_2$ (recall that $y \notin \mathtt{fv}(v) \cup \bigcup_{i=1}^{n} \mathtt{fv}(v_i)$): then, $c_0 \to_{\tilde{\mu}} c_2$.

2. Immediate consequence of Prop. 50.1.
3. Immediate consequence of Prop. 50.1.
4. Immediate consequence of Prop. 50.1. □